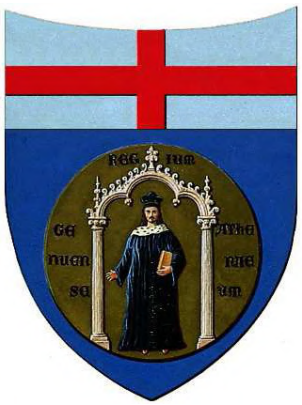
Università degli studi di Genova

Scuola di scienze Matematiche, Fisiche e Naturali
Dottorato in Matematica e Applicazioni - XXXIII ciclo



Doctoral Thesis

Depth-averaged and 3D Finite Volume numerical models for viscous fluids, with application to the simulation of lava flows

Elisa Biagioli

Advisors:
Dr. Mattia de' Michieli Vitturi
Prof. Fabio Di Benedetto

Acknowledgments

From the bottom of my heart, my greatest acknowledgments go to my supervisors Dr. Mattia de' Michieli Vitturi and Prof. Fabio Di Benedetto, reliable, wonderful people and of great value. My highest esteem and gratefulness go to them.

My dissertation has benefited from the collaboration of other people and institutes.

I am also grateful for the precious collaboration of Prof. Einat Lev and Janine Birnbaum from the Lamont-Doherty Earth Observatory at the Columbia University, New York, who hosted my work during a visiting period at their institute and who lent their expertise to produce ad-hoc laboratory tests to support my thesis. In the most critical moment when the world was in lockdown, I was with them in New York City, very far from my home. Their warm and tight support, together with my supervisors, has been crucial, both from a professional and personal point of view.

TanDEM-X WorldDEM data were provided by the German Aerospace Center (DLR) through data proposal DEM_GEOL_1522, PI Nicole Richter, whom I thank for her gentleness and availability to collaborate.

Also, my thesis took great advantage of using the HPC cluster LAKI, thanks to the courtesy of the institute INGV, Pisa Section, that hosts it, particularly to Dr. Luca Nannipieri that facilitated me on this.

I was lucky to be surrounded by the greatest supportive and kind people I could wish for.

First of all, my thanks go to Alessandro Cartatone that technically saved me, my work, and my computer Zeus several times. Alessandro is definitely a spearhead of DiMa and undoubtedly the most likable and best-prepared computer technician ever met.

Then, my warm thanks go to the lovely staff, Eloisa Cilona and Michele Marko Merello Lamperti. A particular thought goes to Paola Bisio, the first person I met at DiMa, and that let me feel welcome at home from that time.

Also, my gratitude goes to my numerous solar colleagues of the past and present. They became friends, pieces of my life and precious memories: Elena Pesce, Francesca Bartolucci, Paola Magrone, Gian Vittorio Luria, Matteo Gardini, CLS, and Greta Coraglia; Francesca Lanteri and Lorenzo Orrù Moglia, Erika Brunetto, Elisa ("the real one") Palezzato, Beyza Yaman, and Ico-Ulderico Fugacci; finally also Sara, Chiara, Silvia, Irene, Daniele, Leonardo and Alessandra from DM, at Pisa. But even out of the University, I enjoyed the invaluable support of friends that enlightened my life: Claudia, Federica, Daria, Nicole, and Valentina, my dear Fox.

I am also very grateful for some professors that let me appreciate both their subjects, their devotion to teaching, and themselves: Dario Bini, Franco Flandoli, Emanuela Sasso, Francesca Morselli, Pietro Di Martino, and, with a bit of repetition, Fabio Di Benedetto.

A fundamental thank goes to Daniele Giordano and our chance encounter that happened many years ago. Daniele let me know about this research sector that I really appreciate and introduced me to Mattia, predicting my Ph.D. No words to describe how much he changed my life over these years. Without him, I would have never embarked on this beautiful journey that has given me enormous joys and satisfaction, and that permitted me to face and overcome many of my fears.

Finally, my last thanks are for my family that supports me and to whom I have a debt of gratitude. In particular, I am infinitely grateful to Lorenzo, who lead me to Genova. He made me fell in love with the city and conducted me to the DiMa. Thanks, from the bottom of my heart, for being this irreplaceable piece of my life!

I dedicate my thesis to the future and the next generation. In particular, I dedicate it to those young people who feel not being enough but have the courage and madness to challenge their fears, giving themselves the opportunity to discover something hidden, precious, different, and new.

> "When the student is ready, the teacher appears" *Japanese proverb.*
>
> "A precious gem is not distinguished from ordinary stones until it is worked by the hands of a master" *Daisaku Ikeda.*

Contents

Introduction

Volcanic eruptions are among the most threatening natural disasters on Earth and overwhelm people living near volcanic buildings. Eruptions may present explosions, with the launch of rocks, gas, and pyroclastic material into the atmosphere, or the propagation of lava from the vent, or both. Every eruption is an unstoppable event, but the effusive phenomena (namely, those for which the lava steadily flows out of the volcano vent) are relatively slow in terms of propagation because the lava front tends to advance at a few hundred meters per hour. So, most of the time, it is possible to respond promptly once the effusive event is underway by calling the civil protection and scientific responders that set up the scenarios of the current event and then prepare evacuation and safety plans. On the other hand, scientists and civil protection can prepare lava flow hazard maps to forecast future scenarios in advance of the eruptions to forecast future scenarios. This is also the only preventive action in the cases of rapid lava flow propagation, a rarer event to be taken into account anyway. For example, the fastest lava flow ever recorded to date is the 1977 eruption of Nyiragongo, which registered speeds of up to 60 kilometers per hour [197] and caused the death of at least 600 people. Therefore, reliable forecastings of lava flow paths require a quantitative description of the effusive phenomenon.

In the 1970s, the traditionally qualitative and observation-oriented field of volcanology started transforming into a quantitative science because modern volcanologists began asking why and how eruptions occur, hence turned to physics and mathematical models to find the answers. The first generations of numerical models for volcanic processes were analytical models based on several simplifications of the real processes (steady-state, 1D, and homogeneous flow), trying to give correlations between the main parameters describing volcanic eruptions. More recently, advancements in the capability to describe physical processes with new mathematical models and the increased computational resources allowed the development of more complex transient and 2D/3D models. Today, computational fluid dynamics (CFD) provides an important and influential approach to studying volcanic eruptions. Now, understanding those natural events involves applying physical laws, numerical techniques for solving the related equations, efficient code implementation, and geological observations. However, even though many volcanic processes have several 3D models describing them with high-resolution physics and thermodynamics (such as the plume models, pyroclastic flows models, and, to a less degree of accuracy, chamber and conduit models), as far as lava flows concern, a smaller number of numerical models has been developed so far. This is mostly due to the complexity, variability, and uncertainties in the physical phenomenon, which has also favored the development of simple stochastic models.

The development of 2D and 3D models of lava flows must account for both fluid dynamics and thermal effects. According to this approach, the physical principles of the conservation of mass, momentum, and energy are the pillars to develop a mathematical model. The numerical code that implements such a model produces numerical simulations

that improve the knowledge of the physical processes governing the dynamics of those natural events. Thus, in addition to the physical laws, a "physics-based" model for lava emplacement must consider:

(i) topography or slope;

(ii) eruptive input conditions such as volumetric effusion rate, vent geometry (point or linear) and effusive temperature;

(iii) thermal boundary conditions at the top and bottom of the lava accounting for insulation, convection, radiation, and conduction;

(iv) physical properties of the lava, like density, thermal conductivity, and viscosity model.

The resulting governing equations form a system of partial differential equations (PDEs) for whom a closed analytical expression of the solutions is not available except for special cases with limited applications. Consequently, numerical methods are more and more employed to compute the approximated solutions of the PDEs. However, since theoretical models and numerical methods cannot capture the entire complexity of lava properties (and particular difficulties arise in considering the viscosity model), simplifications are required (both to make it possible to achieve a solution and speed up codes). Simplifications are acceptable because the properties above impact the lava emplacement in a different way, varying with the distance from the vent but also with time [37, 130]. "Physics-based" (or "deterministic") codes should respect as much as possible the requirements (i - iv) listed above, though this is very hard to do in practice.

Numerical models for lava flow simulations have undergone major developments over the past years, from initially oversimplified models to more complex ones. As already stated, models are distinguished for the deterministic or stochastic approach, the numerical method employed, and the complexity of the physical modeling adopted; hence the associated codes differ for their physical implementations, numerical accuracy, and computational efficiency.

Stochastic models require that the code runs several times with little changes in the parameters; therefore, only codes that are very fast to execute are used. The fastest codes correspond to the most simplified models, which only consider the topography and describe lava as a gravity current following the topography along the steepest slope. DOWNFLOW [83, 252], ELFM [64], VORIS [84], LASZLO [21], MrLavaLoba [66] are examples of probabilistic models. The most simplified versions do not present a stop-criteria for the flow advance and have no description of the temporal evolution.

Channeled models simplify the emplacement process by computing only dynamics that develop in the downslope direction but using complex thermal and viscosity modeling that also accounts for the crystallization process that happens during lava cooling. Codes that implement channeled models, such as FLOWGO [116, 118, 119], are fast to run because they compute 1D dynamics. Nevertheless, a model disadvantage consists of restrictions on the choice of the channel dimensions at the vent to match the effusion rate inputs, and, as already stated, the vent conditions influence the results.

Cellular automata are popular 2D models in which the computational domain is discretized by a 2D grid, and each cell has attributes, such as fluid thickness and temperature. Dynamics is described by the variations of cell properties, which depend on the state of neighboring cells. This model accounts for complex thermal and viscosity models and also

manages to describe lava solidification. Each cellular automata code, such as MAGFLOW [17, 68, 98, 121, 263], SCIARA [5, 7, 8, 60, 61, 62], FLOW FRONT [266, 280], and MOLASSES [49, 70, 155], implements different strategies for the transfer of mass, momentum, and energy between cells.

Depth-averaged models follow the strategy of using depth-averaged variables to obtain a 2D model. This approach is based on the so-called "shallow water approximation", for which the depth (or thickness) of the flow is required to be much smaller than the horizontal scale of the phenomenon of interest. A small aspect ratio of the system implies (by scale analysis of the continuity equation) that vertical velocities are much smaller than horizontal velocities. A way to derive the model is by integrating mass and momentum conservation equations over the fluid depth, from the bottom up to the free surface; otherwise, variational principles are used. De Saint Venant first introduced shallow water approximation in 1864 and Boussinesq in 1872, and their original formulation regarded incompressible and non-viscous fluids. Shallow water models are currently applied to a wide range of geophysical problems for hazard assessment (flood simulations, tsunamis propagation), eventually enriched by additional transport equations such as equations for energy or temperature. This is the case of models for lava flow, where viscosity depends on temperature and thus needs to be accounted for. Costa and Macedonio [55] proposed a depth-averaged model that considers lava cooling and also a temperature-dependent viscosity. More recently, Kelfoun and Vargas [145] followed the depth-averaged model approach too for their lava flow model (creating the numerical scheme VOLCFLOW) with the differences that they adopted a viscoplastic model and used the restrictive hypothesis of isothermal flow. Instead, another recent model that considers temperature evolution and temperature-dependent complex viscosity was proposed in Bernabeu et al. [13], using the RHEOLEF [12, 240] library based on the Finite Element Method. Usually, in the depth-averaged approach, constant vertical profiles are assumed for the variables. These assumptions lead to constant viscosity and temperature along the vertical direction, limiting precise agreement of the viscosity model into the flow dynamics.

The above limit of lack of information about the vertical distribution of variables is overcome by adopting *3D models*. The 3D model approach describes the whole vertical structure of the variables such as velocity, temperature, and viscosity, and precisely these variables are correlated through the viscosity model. In most cases, the 3D description produces multiphase models because it requires solving for the flow of interest and the overlying atmosphere. From a computational point of view, 3D models are surely more expensive to solve than depth-averaged models. Just because of this difficulty, 3D models rose in popularity only recently, when the computational power improved thanks to technological developments and the possibility of parallelizing the execution. FLOW-3D®, COMSOL®, and ANSYS® FLUENT are three examples of CFD commercial software, based on the solution of 3D flow equations on a computational grid, that can apply to lava flows. Among open-source grid-based software, OpenFOAM [137] gained a lot of popularity in the CFD community in the last two decades. Such software includes many solvers to handle complex fluid dynamics problems. Even though no specific solver devoted to lava flow simulations exists, its open-source nature allows the users to modify the code creating one that satisfies their needs by exploiting available solvers and tools. An alternative to grid-based 3D models is the Smoothed Particle Hydrodynamics (SPH) approach, which is a particle-based mesh-free method that follows the Lagrangian modeling to CFD. For example, Zago et al. [282] proposed an SPH model for lava flow simulations that exploits the GPU architecture to parallelize code.

Within this context, this Ph.D. project was initially born from the motivation to contribute to the depth-averaged and 3D modeling of lava flows. Still, we can frame the work done in a broader and more generalist vision. In fact, we developed two models that may be used for generic viscous fluids, and we applied efficient numerical schemes for both cases, as explained in the following.

The new solvers simulate free-surface viscous fluids whose temperature changes are due to radiative, convective, and conductive heat exchanges. A temperature-dependent viscoplastic model is used for the final application to lava flows. Both the models behind the solvers were derived from mass, momentum, and energy conservation laws. Still, one was obtained by following the depth-averaged model approach and the other by the 3D model approach. The numerical schemes adopted in both our models belong to the family of finite volume methods, based on the integral form of the conservation laws. This choice of methods family is fundamental because it allows the creation and propagation of discontinuities in the solutions and enforces the conservation properties of the equations. Codes proposed in this thesis respect all the requirements for numerical methods listed before (*i*–*iv*).

In our work, we propose a depth-averaged model for a viscous fluid in incompressible and laminar regime with an additional transport equation for a scalar quantity varying horizontally and a variable density that depends on such transported quantity. Viscosity and non-constant vertical profiles for the velocity and the transported quantity are assumed, overtaking the classic shallow-water formulation. In fact, the classic formulation bases on several assumptions such as the fact that the vertical pressure distribution is hydrostatic, that the vertical component of the velocity can be neglected, and that the horizontal velocity field can be considered constant with depth because the classic formulation accounts for non-viscous fluids. In fact, when the vertical shear is important, the last assumption is too restrictive, so it must relax, producing a modified momentum equation in which a coefficient, known as the Boussinesq factor, appears in the advective term. The spatial discretization method we employed is a modified version of the central-upwind scheme introduced by Kurganov and Petrova [158] for the classical shallow water equations. This method is based on a semi-discretization of the computational domain, is stable, and, being a high-order method, has a low numerical diffusion. For the temporal discretization, we used an implicit-explicit Runge-Kutta technique [237, 238], that permits an implicit treatment of the stiff terms.

Our 3D model describes the dynamics of two incompressible, viscous, and immiscible fluids, possibly belonging to different phases. Being interested in the final application of lava flows, we also have an equation for energy that models the thermal exchanges between the fluid and the environment. We implemented this model in OpenFOAM, which employs a segregated strategy and the Finite Volume Methods to solve the equations. To deal with the multiphase dynamics, the Volume of Fluid (VoF) technique is used [127], which bases on the Interphase Capturing strategy, and hence a new transport equation for the volume fraction of one phase is added. The challenging effort of maintaining a precise description of the interphase between fluids is solved by using the Multidimensional Universal Limiter for Explicit Solution (MULES) method [196] that implements the Flux-Corrected Transport (FCT) technique [22], proposing a mix of high and low order schemes. The choice of the framework to use for any new numerical code is crucial. In the 3D context, we gave preference to OpenFOAM, thanks to its open-source nature, to its widespread over the community of researchers in computational fluid dynamics, and

because it represents an up-to-date repository for the most efficient and validated numerical schemes existing in literature which are a natural comparison term. This choice has required full knowledge of such a tool.

The original contributions of our work are summarized below.

Concerning the depth-averaged approach, we relaxed the classic assumptions of the shallow water equations of constant density, constant vertical profiles, and no viscosity. In particular, in the whole §2.1 we considered density to depend on a depth-averaged variable, therefore allowing density to vary horizontally. Furthermore, we added a transport equation for a generic scalar quantity with a piecewise linear vertical distribution. As a specific case, we considered the transport equation for the temperature and described its vertical profile in relation to the conductive heat loss to the ground; under these conditions, we obtained a new expression for the heat exchange terms (§2.1.5). We adapted the numerical scheme described before (combining spatial and temporal discretizations from [158] and [237, 238]) to our specific PDE system and proved that it still satisfies important properties like well-balancing (§3.1.1) and positivity preserving for near-dry states (§3.1.2). The implicit discretization leads to a nonlinear system of equations that is solved by the Newton-Raphson procedure. The computation of the Jacobian of the system is hence required by that procedure and clever use of complex arithmetic, borrowed from [248], ensures an accurate computation (§3.3). The resulting numerical scheme was implemented in a Fortran 90 code and has been tested and validated (§3.4) both with literature examples and real applications. In particular, we have used the code with the following tests: Riemann problem with the discontinuous bottom of a non-viscous fluid; dam-break of an isothermal viscous fluid onto flat and inclined bottoms; dam-break of a fluid with low viscosity and temperature-dependent density onto a flat bottom; spreading of an isothermal viscous fluid onto an inclined plane; spreading and cooling of a viscous fluid on a flat bottom; simulation of a lava flow onto Fogo's real topography with plastic temperature-dependent viscosity and radiative, convective, and conductive heat losses. In addition, one of the selected tests was validated with data from a laboratory experiment during the visit at the Lamont-Doherty Earth Observatory (at the Columbia University, New York), performed during the Ph.D. program with the supervision of Prof. Einat Lev and the assistance of Janine Birnbaum. Moreover, some of them come from the paper by Cordonnier et al. [52], which determines recognized benchmarks addressed to lava flow simulation codes to validate their accuracy at different modeling levels.

Thanks to all these simulations, we tested (among others):

- well-balancing and positivity preserving properties of the scheme;
- consequences of using different limiters (i.e. numerical stabilization schemes) and the sensitivity to them;
- the observed differences when adopting constant vertical profiles of the variables or not;
- the effects of the viscosity-related parameters on the final emplacement of a lava flow.

From our work on the depth-averaged model, we published an article [15] (whose content is entirely described in this thesis) where we consider a general viscous fluid: we introduce the vertical profiles for both velocity and transported quantity, accounting also the variable

density case and we present our numerical schemes proving its good properties and testing it with some benchmarks. Another paper is in progress [16], which is more focused on lava application and deals with temperature modeling.

Concerning the 3D multiphase approach, our contribution consists of creating a new solver in the OpenFOAM framework. By modifying an existing solver for two incompressible, immiscible, viscous, and isothermal fluids (interFoam, [69]), we developed the new solver called interThermalRadConvFoam. We added an equation for energy that models radiative and convective heat exchanges (§2.2.2) and implemented the boundary conditions for the heat conduction with the soil. A temperature-dependent viscosity model enriches the model and couples momentum and energy equations. The model was implemented to import DEM files in order to compute simulation over real topography. Dynamic mesh refinement has also been added to the solver to model the interface between fluids more accurately and obtain high accuracy dynamically where necessary. It is also possible to parallelize the computation using the OpenMPI protocol implemented in the software. We compared our simulations with some benchmarks from [52] to evaluate the performances of our model (§4.7): dam-break of an isothermal viscous fluid onto a flat bottom; spreading and cooling of a viscous fluid on a flat bottom. From our work to develop the 3D model, we produced a tutorial that helps to modify interFoam and to obtain our interThermalRadConvFoam (§A), and that will be made available soon for the OpenFOAM user community. Furthermore, we plan to show in a separate paper (subject of future work) the performance of our solver for lava flow simulations.

In this manuscript, the main topics are found in Chapters 2–4 and concern the derivation of the models, the description of numerical schemes that solve them, and the presentation of the simulation results. The bulk of the work is heavily based on four building blocks: the mathematical description of fluid dynamics, the finite volume numerical technique, the OpenFOAM setting for code development, and volcanic applications, which motivated our project. Since this thesis is written trying to be self-supporting, and therefore accessible also to those who are not experts in the field, Chapter 1 collects several details on these four building blocks in order to give the fundamental basis to understand the main topics exposed in the rest of the work. However, an expert reader may decide to skip or come back to it in a second moment since the other chapters present numerous references to the first so that the reader can easily locate what he needs.

Since the modeling point of view in both the depth-averaged and 3D cases is very similar, we gathered them together in Chapter 2, which describes the full derivation of the mathematical models we developed. Thus, the chapter is basically divided into two parts: the first devoted to the depth-averaged model and the second to the 3D multiphase model.

Numerical schemes applied to these models are quite different and hence are described in separate chapters.

Chapter 3 presents the numerical scheme we developed to solve our depth-averaged model, that is, our modified version of the Kurganov and Petrova [158] scheme (for the space discretization), for which we proved the well-balancing and positivity preserving properties, and the implicit-explicit Runge-Kutta method [237, 238] (for the time discretization), which is accompanied by the use of complex arithmetics [248]. Furthermore, results of numerical simulations tested also with literature benchmarks [52] and with a real case are shown.

Chapter 4 first describes the segregated approach that OpenFOAM adopts to solve a

system of PDEs. Then we give a wide overview of the specific solver we developed framed in this context. After, we explore the implementation of the VoF and MULES methods that OpenFOAM uses to treat the multiphase flows. Since OpenFOAM is accompanied by poor documentation, one of our purposes in writing this Chapter 4 was to contribute in this direction by adding specific details for the user. Finally, the results of numerical simulations are shown.

Chapter 5 is dedicated to a final discussion about the results obtained, some conclusive considerations, and the description of possible future developments.

Two Appendices end the manuscript by providing supplementary material. Appendix A contains the Tutorial that helps to reproduce our OpenFOAM code. Appendix B describes the variational derivation of the shallow water equations accompanied by the necessary background notions about the calculus of variations.

In addition to what is presented in this thesis, the research activities carried out during the Ph.D. program led to other collaborations and results. The theoretical investigations on the assumptions on which are based depth-averaged models, for example, contributed to a study on the applicability of such approach to the simulation of tsunami waves generated by landslides at Stromboli island, Italy, carried out in the framework of the collaboration between the Italian Department of Civil Protection (DPC) and the National Institute of Geophysics and Volcanology (INGV). This work is documented in a technical report [80], representing a deliverable of the agreement DPC-INGV, Annex B2 2019-2021, WP12. Studies about lava flow simulations also have been useful for the collaboration with INGV into a science dissemination project that led to the presentation of volcanic eruptions simulations story and to play live simulations of lava flows onto the real topography of Vesuvio inside a sandbox [1].

Chapter 1

Problem overview

This Chapter provides the basis to understand the main contents developed for the project of this Thesis, which mostly occupy Chapters 2, 3, 4. Our project was distributed between fluid dynamics, related computational methods and applications, and this chapter is analogously organized. Furthermore, this manuscript was written trying to be self-supporting, and therefore accessible also to those who are not experts in the field, that is the reason behind the presence of a so detailed Chapter of prerequisites. An expert reader may skip this chapter and go directly to the next. Anyway, the rest of the manuscript has many references to this chapter, so any reader can come back here to see what he needs in a second moment.

Section §1.1 introduces some basis of fluid dynamics, derives the Navier-Stokes equations, and discusses them from an analytic perspective. This Section supports the comprehension of Chapter 2 where we present the derivation of our mathematical models. Moreover, this Section offers the idea of the contexts in which we can insert our work.

Section §1.2 presents Finite Volume Methods, the most diffused family of numerical schemes applied in the computational fluid dynamics, which constitute the core of the numerical models we developed and analyzed in Chapters 3 and 4. In addition, this Section provides the basic concepts about convergence, and some schemes are discussed.

Section §1.3 exposes how the Finite Volume Methods are implemented in the OpenFOAM context. It is important to know this to understand the other methods and strategies implemented in the software that we tackle in Chapter 4.

Section §1.4 presents the phenomenon of effusive eruptions, which is one of the possible applications of our modeling and numerical work and also the initial motivation of our project.

1.1 Physics of fluids

This section presents some background notions about the physics of fluids necessary to define the equations that mathematically describe them.

The word ***fluid*** generally indicates the state of matter, gaseous, liquid, or plasma. Still, its wider and deeper definition is that of a substance that tends to deform (to flow) continually and irrecoverably under applied shear stress or any external force. Even the solid materials may show a fluid behavior when observed on a long time scale. For example, the glaciers formed by the water at the solid-state, even though they appear still at a human glance, actually move very slowly downstream, Figure 1.1a. The rocks constituting the earth mantle behave as elastic solids on short time scales when they

transmit seismic shear waves; still, the mantle behaves like a fluid and exhibits convective motions if it is observed at the geological time scale of 10 Myr (3×10^{14} s), Figure 1.1b. Besides this, by changing the spatial scale of observation, different materials may unexpectedly behave like fluids. The granular solid matters like sand or debris may show a fluid behavior too when observed on a length scale bigger than the granule dimension, Figures 1.1c and 1.1d.

A similar situation happens for multiphase flows, namely those cases where the continuous fluid phase incorporates a dispersed phase: for example, the geysers have bubbles of gas dispersed in hot water, Figure 1.1e, lahars present volcanic pyroclasts dispersed in water, Figure 1.1f. Again, if the composition is observed at a spatial scale larger than the dimensions of the bubbles or the rocks, then the global behavior of these multiphase materials is like a unique fluid.

1.1.1 Continuum hypothesis

The physics of fluid behavior, like the physics of elastic media, is based on the general continuum hypothesis. This hypothesis requires that quantities like density, temperature or velocity, are defined everywhere, continuously, and at "points" of infinitesimal volumes containing a statistically meaningful number of molecules (or granules) so that such quantities represent averages, independent of microscopic molecular fluctuations. The "points" are called ***material element*** in continuum mechanics or ***fluid parcel*** in fluid mechanics. Their dimensions must be bigger than the average distance between the molecules (or granules) length scale but smaller than the characteristic length scale of the system. The assumption of the continuum hypothesis might lead to results that are not of the desired accuracy in some circumstances. For example, for flows with supersonic speed or in molecular flows on a nanoscale, the continuum hypothesis fails [109], and the problems are solved by passing to the statistical mechanics. Despite this, under the right circumstances, the continuum hypothesis produces extremely accurate results.

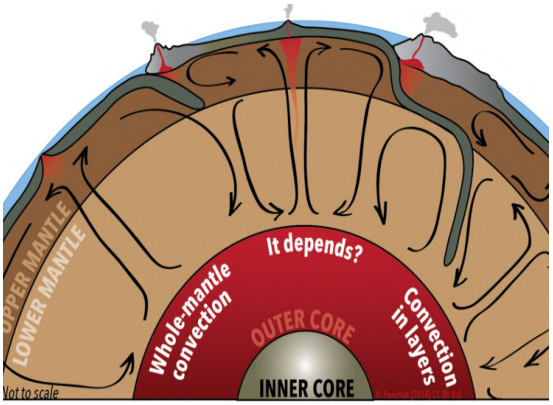
1.1.2 Conservation laws

In general, the equations that describe the motion of the fluid are derived from the following three *conservation laws*:

- ***Conservation of mass***: the mass of a closed system (in the sense of a completely isolated system) remains constant in time;
- ***Conservation of momentum***: the rate of change of the momentum of a closed system is equal to the net forces acting on it (Newton's second law);
- ***Conservation of energy***: the total amount of energy in a closed and isolated system remains constant over time.

The resulting equations are called *governing equations* (or *balance equations*) and express, in a mathematical framework, the general physical principles that underlie the processes of continuous bodies. Thus, the movement understood as the evolution of a continuous body is physically admissible only if it satisfies, at each instant, the appropriate equations.

(a) *Glacier.* The Perito Moreno Glacier at Los Glaciares National Park, Argentina. National Geographic.



(b) *Models of mantle convection.* University of Saskatchewan.

(c) *Sand.* From: Learning Geology.

(d) *Debris.* From: Learning Geology.

(e) *Geyser.* Castle Geyser in Yellowstone National Park. From: World Atlas.

(f) *Lahar (mudflows).* Lahars in a river valley Soufrière Hills Volcano, Montserrat. From: BGS.

Figure 1.1: Different matters that exhibit a fluid behavior. *Top row*: solids that have a fluid-like dynamics in a long time scale. *Middle row*: granular solid matters that show a fluid behavior if observed on a length scale much bigger than the granule dimension. *Bottom row*: multiphase matters, similarly to the previous examples, act like fluid if observed on a length scale bigger than the dimension of the particles dispersed.

1.1.3 Lagrangian and Eulerian approach

To study the behavior of fluids and transpose in mathematical terms the conservation laws, two different points of view are adopted: a *Lagrangian approach* and a *Eulerian approach*.

- ***Lagrangian approach***: it is a way of looking at fluid motion where the observer follows an individual fluid parcel as it moves through space and time; therefore, the individual fluid particles are tracked as they move and deform through the flow field, as in Figure 1.2, left panel.
- ***Eulerian approach***: it is a way of looking at the fluid motion by focusing on a fixed location in space. The characteristics of the flow are evaluated for an imaginary and fixed *control volume*, see Figure 1.2, right panel.

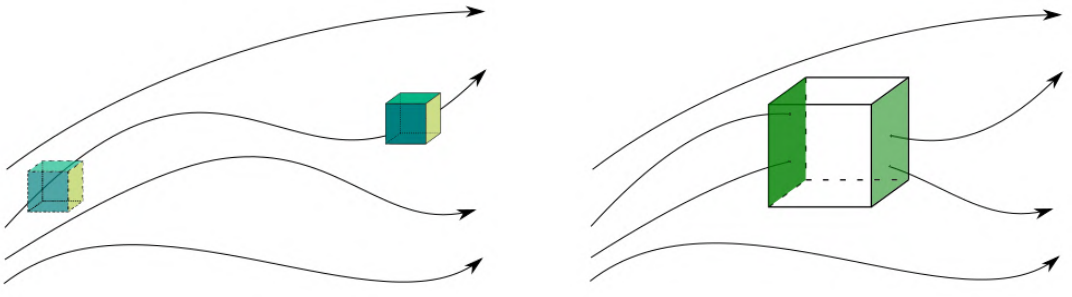


Figure 1.2: Comparison between Lagrangian approach (picture on the left), where the parcel follows the flow, and Eulerian approach (picture on the right), where a control volume is fixed and we observe what happens inside it.

An example of a Lagrangian description is given by a marine survey buoy that, free to move with the ocean surface currents, provides the meteo-marine data such as temperature, salinity, pollution, and the current velocity. Whereas when the buoy is anchored gives an Eulerian description because it records the fluid parameters related only by its fixed position.

In the next paragraph, we will see that these two different approaches are associated with two "different" temporal derivatives, and the *material derivative* is introduced. In §1.1.4 we adopt the *Eulerian approach* to derive the transport equation for a generic conservative quantity. In §1.1.5 we adopt the Lagrangian point of view to derive the governing equations related to the conservation laws of §1.1.2.

Material derivative. In the *Eulerian* formalism, the spatial coordinates $\mathbf{x} = (x, y, z)$ are the independent variables and they refer to a fixed reference system. Each physical quantity q related to the flow, may it be a scalar (such as density, pressure, and temperature) or a vector quantity (like velocity), is defined in each point $\mathbf{x}$ and for every time t and it constitutes a scalar field or vector field respectively, hence $q = q(\mathbf{x}, t)$, $\forall t, \forall \mathbf{x}$.

In the *Lagrangian* framework, the focus is on a single fluid parcel which is followed in the flow. The position of a fluid parcel initially located in $\mathbf{r}_0 = (x_0, y_0, z_0)$ is a function of time expressed as

$$\mathbf{x}_{\mathbf{r}_0}(t) = (x_{\mathbf{r}_0}(t), y_{\mathbf{r}_0}(t), z_{\mathbf{r}_0}(t)) \,.$$

The velocity of a fluid parcel is the time derivative of its position, namely

$$\mathbf{u}_{\mathbf{r}_0}(t) = \frac{d\mathbf{x}_{\mathbf{r}_0}(t)}{dt} = \left(\frac{dx_{\mathbf{r}_0}(t)}{dt}, \frac{dy_{\mathbf{r}_0}(t)}{dt}, \frac{dz_{\mathbf{r}_0}(t)}{dt} \right). \tag{1.1}$$

For every time t, the velocity field in the Eulerian formalism $\mathbf{u} = (u, v, w)$ can be expressed in term of the fluid parcels velocity

$$\mathbf{u}(\mathbf{x}, t) = \mathbf{u}_{\mathbf{r}_0}(t) \tag{1.2}$$

where $\forall t, \forall \mathbf{x}$, it exists a fluid parcel initially located in $\mathbf{r}_0$ such that at time t its position is exactly $\mathbf{x} = \mathbf{x}_{\mathbf{r}_0}(t)$. Therefore the fluid parcels velocities constitute the entire velocity field.

The Lagrangian formalism allows to describe how the quantity represented by the field $q(t, \mathbf{x})$ evolves on each line associated with the motion of a fluid parcel initially located at $\mathbf{r}_0$:

$$q_{\mathbf{r}_0}(t) = q(\mathbf{x}_{\mathbf{r}_0}(t), t). \tag{1.3}$$

The time derivative of the Lagrangian representation of q, namely of $q_{\mathbf{r}_0}$, corresponds to the computation of the variations of q along the history of a fluid parcel

$$\frac{dq_{\mathbf{r}_0}}{dt}(t).$$

From Eq. (1.3) and by reminding Eqs. (1.1) and (1.2), the previous derivative is computed with the chain rule and it might be expressed as

$$\begin{aligned} \frac{dq_{\mathbf{r}_0}}{dt}(t) &= \frac{\partial q}{\partial t}(\mathbf{x}, t) + \frac{\partial q}{\partial x}(\mathbf{x}, t)\frac{dx}{dt} + \frac{\partial q}{\partial y}(\mathbf{x}, t)\frac{dy}{dt} + \frac{\partial q}{\partial z}(\mathbf{x}, t)\frac{dz}{dt} \\ &= \frac{\partial q}{\partial t}(\mathbf{x}, t) + u\frac{\partial q}{\partial x}(\mathbf{x}, t) + v\frac{\partial q}{\partial y}(\mathbf{x}, t) + w\frac{\partial q}{\partial z}(\mathbf{x}, t) \\ &= \frac{\partial q}{\partial t}(\mathbf{x}, t) + (\mathbf{u} \cdot \nabla) q(\mathbf{x}, t), \end{aligned}$$

where $\mathbf{x} = \mathbf{x}_{\mathbf{r}_0}(t)$. The Lagrangian derivative is also called ***material derivative*** (or particle derivative) of the field q, and the following notation is used

$$\frac{Dq}{Dt}(\mathbf{x}, t) = \frac{\partial q}{\partial t}(\mathbf{x}, t) + (\mathbf{u} \cdot \nabla) q(\mathbf{x}, t), \tag{1.4}$$

The first term on the right side, $\partial q / \partial t$, is the local rate of change (the unsteady term), and it is null for steady flows, that is, for those flows such that the flow properties at every point fixed in space do not depend upon time. For example, a river might be an example of steady flow, with some approximations, because the velocity and pressure fields are constant in time at a fixed location, even though they may change in space (for example, because of narrowing or widening). The second term on the right-hand side is called convective derivative, and it is associated with changes in the space of the quantity of interest. Here the gradient of q is computed at a point $\mathbf{x}$, fixed in space and time, thus can be seen as a gradient computed in the Eulerian framework. With this in mind, we can see that the terms of the material derivative can be computed in the more comfortable representation of the Eulerian formalism, even if it represents the rate of change observed by a fluid particle that moves with the fluid. Thus, it is seen as a link between the two formalisms, Lagrangian and Eulerian.

In the next two paragraphs, we adopt the Eulerian and Lagrangian approaches to show the different derivation of equations from conservation principles. In particular, the Eulerian approach is adopted in §1.1.4 to derive the transport equation for a generic conservative quantity, as would be the derivation of the mass conservation equation. On the other hand, the Lagrangian approach is instead employed in §1.1.5 to derive all the governing equations related to the conservation laws stated in §1.1.2, leading to the Navier-Stokes equations system.

1.1.4 Transport equation

In engineering, physics, and chemistry, the study of transport phenomena concerns the exchange of mass, momentum, and energy (but also other properties of the fluid) between observed and studied systems. Mass, momentum, and heat transport all share a very similar mathematical framework (as PDEs). The parallels between their PDEs are exploited in the study of transport phenomena to draw mathematical connections that often provide useful tools in analyzing one field that is directly derived from the others. While it draws its theoretical foundation from principles in several fields, most of the fundamental transport theory is a restatement of the basic conservation principles given in §1.1.2. Thus, we see the example of a transport equation derivation from a generic conservative principle in the Eulerian framework.

Let us consider a fluid moving with velocity $\mathbf{u}(\mathbf{x}, t)$ and a property $q(\mathbf{x}, t)$, which may be density or any other quantity for which there exists a conservation law. For example, in a fluid with sand suspension, q may represent the suspension's volume fraction; instead, when modeling the sea and oceans, q might be the volumetric fraction of salt dissolved. By adopting the *Eulerian approach*, we focus on an *imaginary control volume* V, fixed in the space, immersed in the fluid as in Figure 1.3. We call $\mathcal{Q}_V(t)$ the total amount of the property q in the control volume V at time t, namely

$$\mathcal{Q}_V(t) := \int_V q(\mathbf{x}, t) dV, \tag{1.5}$$

and we want to express the variation of $\mathcal{Q}_V$ in the control volume over time assuming that it is governed by the conservation principle that states:

> *The rate of change of the property $\mathcal{Q}_V$ described by q inside a fixed control volume V equals the net flux through the surface that constitutes the boundary ∂V of the control volume.*

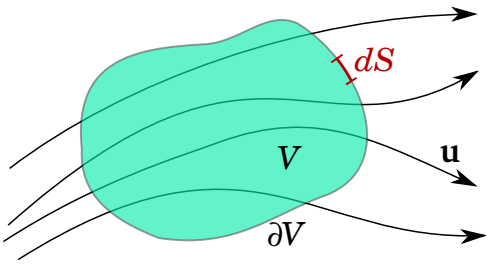


Figure 1.3: Flux through a fixed volume V.

The variation of $\mathcal{Q}_V$ in a time interval Δt (represented by $\Delta\mathcal{Q}_V = \mathcal{Q}_V(t + \Delta t) - \mathcal{Q}_V(t)$) is due to the amount of q that enters and exits from the control volume. We quantify

the amount of property lost that exits from an infinitesimal surface area dS of ∂V in the interval of time Δt. If the fluid moves with velocity $\mathbf{u}$ and if $\hat{\mathbf{n}}$ is the unit outward vector orthogonal to the infinitesimal element dS, see Figure 1.4, then $\mathbf{u} \cdot \hat{\mathbf{n}}\, dS\Delta t$ is the volume occupied by the molecules passing through dS in the time interval Δt. As a consequence, the property q that passes through the infinitesimal surface element is $-q\mathbf{u} \cdot \hat{\mathbf{n}}\, dS\Delta t$. By

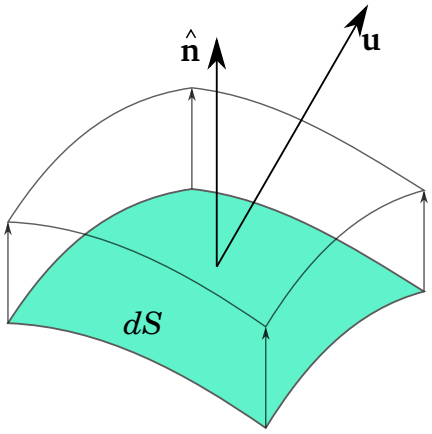


Figure 1.4: Volume swept by the outgoing molecules from the infinitesimal surface element dS in the time interval Δt between t and $t + \Delta t$.

integrating all over the surface ∂V, we obtain that the rate of variation of $\mathcal{Q}_V$ occurred in the time interval Δt is

$$\frac{\Delta \mathcal{Q}_V}{\Delta t} = \frac{\mathcal{Q}_V(t + \Delta t) - \mathcal{Q}_V(t)}{\Delta t} = -\int_{\partial V} q\mathbf{u} \cdot \hat{\mathbf{n}}\, dS.$$

Notice that the surface integral term is the flux passing through the surface ∂V, namely the flux of the vector $q\mathbf{u}$. Passing at the limit $\Delta t \to 0$, the expression results in

$$\frac{d\mathcal{Q}_V}{dt} = -\int_{\partial V} q\mathbf{u} \cdot \hat{\mathbf{n}}\, dS. \tag{1.6}$$

From Eq. (1.5), the rate of change of $\mathcal{Q}_V$ in the control volume is

$$\frac{d\mathcal{Q}_V}{dt} = \frac{d}{dt}\left(\int_V q dV\right) = \int_V \frac{\partial q}{\partial t} dV, \tag{1.7}$$

where the time derivative and the integral operators invert the order because V does not depend on time. Whereas, we use the *Gauss Theorem* (that follows) to express the surface integral in (1.6) over ∂V as a volume integral over V. The *Gauss theorem* alternate names are *divergence theorem* and *Ostrogradskij theorem* because the first proof of such theorem was provided by M. Ostrogradskij [202].

Theorem 1.1 (Gauss Theorem)**.** *Suppose V is a subset of $\mathbb{R}^n$ which is compact and has a piecewise smooth boundary ∂V. If $\mathbf{F}$ is a continuously differentiable vector field defined on a neighborhood of V, then:*

$$\int_V \nabla \cdot \mathbf{F} dV = \int_{\partial V} \mathbf{F} \cdot \hat{\mathbf{n}} dS.$$

Thanks to the Gauss theorem, we obtain that:

$$-\int_{\partial V}(q\mathbf{u})\cdot\hat{\mathbf{n}}\,dS=-\int_{V}\nabla\cdot(q\mathbf{u})dV. \tag{1.8}$$

From Eq. (1.6), the two quantities in Eqs. (1.7) and (1.8) must be equal, then the integral form of the conservation law descends and, by moving every term on the left, it becomes as follows:

$$\int_{V}\left[\frac{\partial q}{\partial t}+\nabla\cdot(q\mathbf{u})\right]dV=0. \tag{1.9}$$

It remains to discuss the role of pressure p. For that, we assume a Newtonian rheology. If it is the thermodynamic pressure, then the deviatoric stress $\mathbf{d}$ follows Eq. (1.29) or (1.28) and there is a *state equation* that relates pressure with the other state variables. For instance, in the case of a perfect gas the following relation holds:

As this equality is true for every control volume V, if the integrand is continuous, thanks to the *fundamental lemma of calculus of variations* [100, Lemma 1, page 9], we obtain the differential form of the transport equation

$$\frac{\partial q}{\partial t}+\nabla\cdot(q\mathbf{u})=0$$

Note that the integral form of the equation for the conservation of q may be used without restrictions on the regularity of the functions: even in the case of discontinuous solutions, all the integral form terms of the equation are well defined. Conversely, the differential form requires q to be differentiable in both space and time and u to be differentiable with respect to space.

Considering the density ρ as the property q, we have that its integral over the control volume is simply the volume mass. The previous derivation is still valid for $q=\rho$, and it gives the mass conservation equation (which is commonly called *continuity equation*, and we refer to the equation in both ways in the rest of the work). For the momentum conservation, the property q to be transported is the momentum density $\rho\mathbf{u}$, which volume integral over the control volume is the total momentum of the volume. Since momentum is a vectorial quantity, the equations are written for the conservation of each component. Whereas for energy conservation, the conservative quantity q is the density of total energy E, and its volume integral is the total energy possessed by the volume. The rigorous derivation of momentum and energy equations is a little more complicated than the mass equation, and we skip it in the Eulerian approach. However, we derive the equations completely in the Lagrangian framework in the next section. Anyway, we anticipate that the derivation of the conservative equations produces equations such as the following generic transport equation:

$$\frac{\partial q}{\partial t}+\nabla\cdot f(q)=S(q),$$

where the function f is called *flux function* and is related with the flux through the surface of the property q, whereas the function S contains the source terms. If the conservative property is a vectorial quantity, we have the divergence operator applied to a matrix.

1.1.5 Navier-Stokes equations

In this section, we use the Lagrangian approach (described in §1.1.3) to derive the governing equations corresponding to the conservation laws for the mass, momentum, and

energy. Conservation laws, as introduced in §1.1.2, are already formalized according to the Lagrangian formulation because they refer to a closed system (in the sense of a completely isolated system) that we may assimilate to whatever fluid parcel. For this reason, it seems that the Lagrangian derivation is "more natural" than the Eulerian one. The governing equations that we obtain for the mass, momentum, and energy conservation are the so-called *Navier-Stokes equations*. Historically, only the momentum equation was entitled to Navier-Stokes, but in the modern context, instead, this name is given to the whole system. The power of the Navier-Stokes equations is to describe the motion of whatever viscous fluid mathematically through a system of PDEs. Under different assumptions, the Navier-Stokes equations may change and simplify their expressions. For instance, in §1.1.5.6 we analyze what happens in the incompressible case. Besides the derivation of the equations, we examine some viscosity models in §1.1.5.7 and, by the introduction of the well-known Reynolds number, we present the classification of laminar and turbulent flows in §1.1.5.8.

In the next paragraphs, we consider a fluid parcel as it moves with the flow, and we denote as V_t the spatial region occupied by the parcel at time t.

1.1.5.1 Derivation of the mass conservation equation

Being $\rho(\mathbf{x}, t)$ the mass density function, the mass inside the fluid parcel at the time t is defined by the integral

$$M(t) := \int_{V_t} \rho(\mathbf{x}, t) dV. \tag{1.10}$$

From the mass conservation law, as stated in §1.1.2, the mass $M(t)$ is constant for each t hence its time derivative must be null

$$\frac{dM(t)}{dt} = 0. \tag{1.11}$$

In order to obtain the mass governing equation, we introduce an important result of Reynolds [225] which rules how to move the time derivative from out to inside an integral defined over a time-dependent region.

Theorem 1.2 (Reynolds transport theorem I)**.** *Consider integrating a function $f = f(\mathbf{x}, t)$ over the time-dependent region V_t; then taking its derivative with respect to time results in*

$$\begin{aligned} \frac{d}{dt}\int_{V_t} f dV &= \int_{V_t} \left[\frac{Df}{Dt} + f(\nabla \cdot \mathbf{u})\right] dV \\ &\overset{(1.4)}{=} \int_{V_t} \left[\frac{\partial f}{\partial t} + (\mathbf{u} \cdot \nabla) f + f(\nabla \cdot \mathbf{u})\right] dV \\ &= \int_{V_t} \left[\frac{\partial f}{\partial t} + \nabla \cdot (f\mathbf{u})\right] dV. \end{aligned} \tag{1.12}$$

By using the Reynolds transport theorem I we obtain that for each fluid parcel

$$\begin{aligned} \frac{dM(t)}{dt} &\overset{(1.10)}{=} \frac{d}{dt}\int_{V_t} \rho(\mathbf{x}, t) dV \\ &\overset{(1.12)}{=} \int_{V_t} \left[\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho\mathbf{u})\right] dV \\ &\overset{(1.11)}{=} 0. \end{aligned}$$

Notice that, if the integrand function is continuous, since the previous equation must be valid for each control volume considered, then the integrand function itself results to be null, namely it descends that

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{1.13}$$

that is the differential form of the mass conservation equation.

1.1.5.2 Derivation of the momentum conservation equation

The momentum of a fluid parcel occupying the volume V_t at the time t is defined as the integral of the momentum density $\rho\mathbf{u}$

$$\mathbf{P}(t) := \int_{V_t} \rho(\mathbf{x}, t)\mathbf{u}(\mathbf{x}, t) dV. \tag{1.14}$$

According with the conservation of momentum principle, at each time t, the rate of change of momentum $\mathbf{P}(t)$ is equal to the sum of the forces acting on the fluid parcel

$$\frac{d\mathbf{P}(t)}{dt} = \mathbf{R}(t). \tag{1.15}$$

Before deriving the differential equation for the momentum conservation, we introduce a second result useful in the next computations.

Theorem 1.3 (Reynolds transport theorem II)**.** *If the continuity equation (1.13) is valid, integrating the product $\rho(\mathbf{x}, t)f(\mathbf{x}, t)$ over the time-dependent region V_t and taking its derivative with respect to time results in*

$$\frac{d}{dt}\int_{V_t} \rho f dV = \int_{V_t} \rho \frac{Df}{Dt} dV. \tag{1.16}$$

Using the Reynolds transport theorem II, the rate of change of the momentum is expressed as follows

$$\frac{d\mathbf{P}(t)}{dt} \overset{(1.14)}{=} \frac{d}{dt}\int_{V_t} \rho\mathbf{u} dV \overset{(1.16)}{=} \int_{V_t} \rho \frac{D\mathbf{u}}{Dt} dV.$$

By using the definition of material derivative and the continuity equation, the integrand of the previous equation might be rewritten as

$$\begin{aligned} \rho\frac{D\mathbf{u}}{Dt} &\overset{(1.4)}{=} \rho\frac{\partial \mathbf{u}}{\partial t} + \rho\mathbf{u} \cdot (\nabla\mathbf{u}) \\ &= \rho\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\rho\mathbf{u}\mathbf{u}^T) - \mathbf{u}\nabla \cdot (\rho\mathbf{u}) \\ &\overset{(1.13)}{=} \rho\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\rho\mathbf{u}\mathbf{u}^T) + \mathbf{u}\frac{\partial \rho}{\partial t} \\ &= \frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla \cdot (\rho\mathbf{u}\mathbf{u}^T). \end{aligned} \tag{1.17}$$

As a consequence the rate of change of momentum results to be

$$\frac{d\mathbf{P}(t)}{dt} = \int_{V_t} \left[\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla \cdot (\rho\mathbf{u}\mathbf{u}^T) \right] dV. \tag{1.18}$$

The forces acting on the fluid parcel are of two types, bulk and contact forces which sum respectively as $\mathbf{F}^{(b)}(t)$ and $\mathbf{F}^{(c)}(t)$. A bulk force is, for example, a gravitational field, and we assume for simplicity that it is the only one bulk force acting on the system (other bulk forces may be prescribed otherwise). Instead, the contact forces act on the fluid parcel through its boundary because of the interaction with the neighbor parcels or with the external environment; such forces are entirely described by the stress tensor $\boldsymbol{\sigma}$ (whose expression will be described in the next §1.1.5.3). In the following equation, we give the expression of the net external forces at the right hand side of (1.15) and use the Gauss theorem to pass from a surface integral to a volumetric integral:

$$\begin{aligned}\mathbf{R}(t) &= \mathbf{F}^{(b)}(t) + \mathbf{F}^{(c)}(t) \\ &= \int_{V_t} \rho \mathbf{g} dV + \int_{\partial V_t} \mathbf{n} \cdot \boldsymbol{\sigma} dS \\ &= \int_{V_t} \rho \mathbf{g} dV + \int_{V_t} \nabla \cdot \boldsymbol{\sigma} dV,\end{aligned} \tag{1.19}$$

where $\mathbf{n}$ is the unit outward vector normal to the volume surface.

Reassembling the results (1.18) and (1.19) together in the initial Eq. (1.15), we get the integral form of the momentum conservative equation

$$\int_{V_t} \left[\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}^T) - \rho \mathbf{g} - \nabla \cdot \boldsymbol{\sigma} \right] dV = 0.$$

Also, as in the continuity equation case, the condition above must be respected for every parcel V. Therefore, the integrand itself must be null so that the differential form of the momentum conservation equation descends:

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}^T) = \rho \mathbf{g} + \nabla \cdot \boldsymbol{\sigma}. \tag{1.20}$$

In the next section, we analyze the tensor $\boldsymbol{\sigma}$ that contributes to the contact forces acting on the boundary surface of the fluid parcels.

1.1.5.3 Stress tensor, pressure and viscosity

The stress forces acting on a fluid surface are a combination of normal stress, which may be compression or tension, as represented in Figures 1.5a and 1.5b, and tangential stress (also called *shear stress*), see Figure 1.5c. The *stress tensor* $\boldsymbol{\sigma}$ is a second-order tensor that describes these stress forces. Each element σ_{ij} denotes the i-component of the force per unit area acting on a plane surface normal to the j-direction. The diagonal entries model the normal stresses and the non-diagonal elements the shear stresses.

From a mathematical point of view, $\boldsymbol{\sigma}$ is a symmetric matrix; hence it always exists a local system of orthogonal axes $\mathcal{B}$ that brings the stress tensor to a diagonal form $\boldsymbol{\sigma}_{\mathcal{B}}$. The matrix eigenvalues are the so-called *principal stresses*, σ'_{11}, σ'_{22}, σ'_{33}, and each of them corresponds to a tension or to a compression, depending on the sign. According to this, each fluid parcel undergoes a superposition of tensions/compressions along with the three local orthogonal directions $\mathcal{B}$. Reminding that the trace of a matrix, denoted as t, is invariant for a change of basis, namely

$$t := \mathrm{tr}(\boldsymbol{\sigma}) = \sum_{i=1}^{3} \sigma_{ii} = \sigma'_{11} + \sigma'_{22} + \sigma'_{33} = \mathrm{tr}(\boldsymbol{\sigma}_{\mathcal{B}}),$$

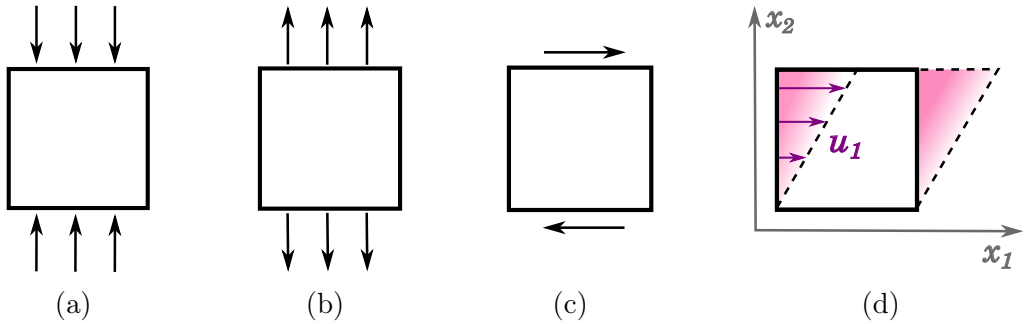


Figure 1.5: Normal (a,b) and tangential (c) stress forces acting on the surface normal to x_1 and the flow consequent to the shear stress (d).

then the stress tensor, expressed with respect to $\mathcal{B}$, may be split as follows:

$$\boldsymbol{\sigma}_{\mathcal{B}} = \begin{bmatrix} \sigma'_{11} & 0 & 0 \\ 0 & \sigma'_{22} & 0 \\ 0 & 0 & \sigma'_{33} \end{bmatrix} = \begin{bmatrix} \frac{1}{3}t & 0 & 0 \\ 0 & \frac{1}{3}t & 0 \\ 0 & 0 & \frac{1}{3}t \end{bmatrix} + \begin{bmatrix} \sigma'_{11} - \frac{1}{3}t & 0 & 0 \\ 0 & \sigma'_{22} - \frac{1}{3}t & 0 \\ 0 & 0 & \sigma'_{33} - \frac{1}{3}t \end{bmatrix}.$$

The first tensor has spherical symmetry (it is *isotropic*) that corresponds to a uniform compression or uniform tension; hence we can see it as a tensor representation of *pressure*. As a consequence of the isotropic property of pressure, it tends to change the volume of the fluid parcel at which they are applied as represented in Figures 1.6a and 1.6b. The second tensor has a null trace, so the normal stresses represented by that tensor must be at least one compression and one tension, with the consequence that the fluid parcel deforms, see Figure 1.6c.

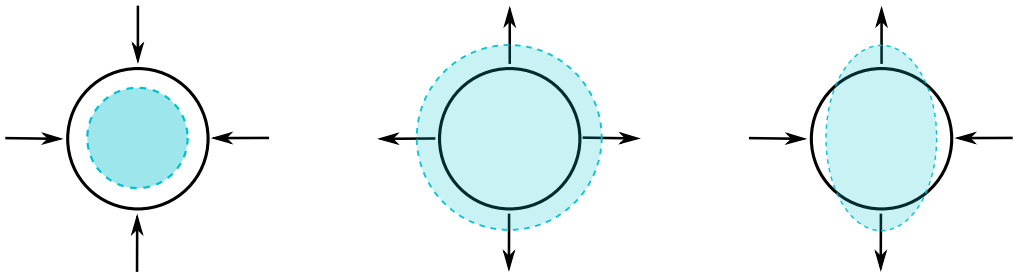


Figure 1.6: Different combination of compressions and tensions acting on a spherical fluid parcel.

If a fluid is *at rest*, it does not move, so it does not deform, and its volume does not change. The absence of deformation translates that the second tensor is null. This also means that the principal stresses are the same, that is $\sigma'_{11} = \sigma'_{22} = \sigma'_{33} = \frac{1}{3}t$ at all points in the fluid; therefore, the stress tensor in a fluid at rest is everywhere isotropic, all orthogonal axes of reference are principal axes for the stress tensor and only normal stresses act. Moreover, the fluids at rest are normally in a state of compression, so it is convenient to write the stress tensor as

$$\boldsymbol{\sigma} = -p\mathbf{I}, \tag{1.21}$$

$$p = -\frac{1}{3}t, \tag{1.22}$$

where we introduce p, that we call *pressure* (and, since the fluid is at rest, we could see that as the hydrostatic pressure, which may be a function of position), notice that such

a definition of pressure is purely mechanical. Despite it, for a fluid at rest, the pressure p coincides with the thermodynamic pressure that is a state variable (which may depend on the temperature and density, for example). Underlining this difference, we introduce the notations p_{mech} and p_{th} to distinguish in general the mechanical and thermodynamic pressure at equilibrium. Hence, for a fluid at rest, the following equation holds

$$p = p_{mech} = p_{th}.$$

However, for a moving fluid, the mechanical and the thermodynamic pressure may not coincide, and we deal with it in detail below.

When the fluid is not at rest, the previous observations and results do no hold. Despite this, it is still useful to have the analog of the static pressure also for a moving fluid, measuring the local intensity of the "squeezing" of the fluid. Then we continue to call *pressure* the isotropic part of the stress tensor, whereas the remaining non-isotropic term $\mathbf{d}$ is named *deviatoric stress tensor* or *viscous stress tensor*:

$$\boldsymbol{\sigma} = -p\mathbf{I} + \mathbf{d}. \tag{1.23}$$

The deviatoric stress tensor is present only in the case of motion; in fact, it depends on the velocity distribution or, more precisely, on the deviation from the uniformity of such distribution. From this observation descends that the main parameter of the deviatoric stress tensor is the velocity gradient. In order to clarify the last statement, consider pure shear stress, as represented in Figure 1.5c, that acts on a rectangular portion of a viscous fluid and makes it move. One may imagine as if a series of parallel planes move one over the other, with the upper layers that run faster than those underlying with a resulting configuration similar to that represented in Figure 1.5d. The velocity shows in this example a non-zero gradient $\dfrac{\partial u_1}{\partial x_2} \neq 0$. The difference of velocity between two contiguous layers is named *shear rate* (or *strain rate* when dealing with the deformation of solids), and the symmetric tensor formalizes it

$$\mathbf{E} := \frac{1}{2}\left[\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right] \tag{1.24}$$

that is called *shear rate tensor*.

Aiming to find a relationship between the stress tensor $\boldsymbol{\sigma}$ and the velocity derivatives $\partial u_i/\partial x_j$, some assumptions are made on the fluid properties and on the tensor:

1. the fluid is assumed homogeneous (its physical characteristics are constant in space) so that the spatial variations of the stress tensor depend only on the spatial variations of the velocity field;

2. the fluid is assumed isotropic, namely without preferred direction as far as the relationship between stress and rate of shear concerns. This assumption is not a great restriction, in fact the isotropic structure characterizes all gases and most liquids; non-isotropic liquids are, for example, solutions containing very long chain-like molecules because they present some directional preferences due to the alignment of such molecules;

3. the stress tensor $\boldsymbol{\sigma}$ depends only on the velocity derivatives, hence on the spatial distribution of the velocity field;

4. the relationship between the stress tensor and the velocity derivatives is linear

$$\boldsymbol{\sigma}_{ij} = \sum_{k,l=1}^{3} a_{ijkl} \frac{\partial u_k}{\partial x_l}. \tag{1.25}$$

Fluids that respect the conditions 1-4 exhibit a *Newtonian* behavior. Being $\boldsymbol{\sigma}$ symmetric, the tensor $\mathbf{a}$ has a symmetry with respect to i and j. Without entering in details of further considerations and calculations (refer to the book [9], §3.3 for further details), the following expression of the stress tensor Eq. (1.23) for a Newtonian fluid holds:

$$\boldsymbol{\sigma} = -p\mathbf{I} + \underbrace{2\mu\mathbf{E} + \lambda(\nabla \cdot \mathbf{u})\mathbf{I}}_{\mathbf{d}}, \tag{1.26}$$

where the coefficient μ relates the shear rate tensor to the stress and is called *dynamic viscosity*, whereas the coefficient λ is called *bulk viscosity* (also *volume viscosity*) and is related with the damping associated with volumetric straining. It remains to determine what the pressure p means, and there are two possible definitions.

Case A. We may adopt a mechanical point of view assuming that pressure $p = p_{mech}$ and respects the condition $p = -\frac{1}{3}\mathrm{tr}(\boldsymbol{\sigma})$, as it is for a fluid at rest. From this, it descends a relation between the viscosity coefficients:

$$\mathrm{tr}(\boldsymbol{\sigma}) = -3p + 2\mu\mathrm{tr}(\mathbf{E}) + 3\lambda(\nabla \cdot \mathbf{u}) \overset{(1.24)}{=} -3p + 2\mu(\nabla \cdot \mathbf{u}) + 3\lambda(\nabla \cdot \mathbf{u}) \overset{(1.22)}{\Longrightarrow} \lambda = -\frac{2}{3}\mu,$$

and hence the stress tensor is rewritten as:

$$\boldsymbol{\sigma} = -p_{mech}\mathbf{I} + \underbrace{2\mu\mathbf{E} - \frac{2}{3}\mu(\nabla \cdot \mathbf{u})\mathbf{I}}_{\mathbf{d}}. \tag{1.27}$$

Case B. On the other hand, if in Eq. (1.26) we define p as the thermodynamic equilibrium pressure $p = p_{th}$ (in the cases where the thermodynamic is involved) the stress tensor is simply

$$\boldsymbol{\sigma} = -p_{th}\mathbf{I} + \underbrace{2\mu\mathbf{E} + \lambda(\nabla \cdot \mathbf{u})\mathbf{I}}_{\mathbf{d}}. \tag{1.28}$$

To write the expression of the stress tensor in a way that links the mechanics and the thermodynamics, we use the equilibrium thermodynamic pressure p_{th} that in this case differs from the mechanic pressure of a quantity proportional to the divergence of the velocity field, hence in an isotropic medium, the relation is

$$p = p_{mech} = p_{th} - \zeta \nabla \cdot \mathbf{u},$$

where ζ is a proportionality coefficient depending on the dynamic and bulk viscosity, as described in the following. Therefore, the stress tensor of Eq. (1.27) becomes:

$$\boldsymbol{\sigma} = -p_{th}\mathbf{I} + \underbrace{2\mu\mathbf{E} + \left(\zeta - \frac{2}{3}\mu\right)(\nabla \cdot \mathbf{u})\mathbf{I}}_{\mathbf{d}}. \tag{1.29}$$

In both approaches, the equations we obtain when considering the thermodynamic pressure, Eqs. (1.29) and (1.28), are the same and linked by the relation

$$\lambda = \zeta - \frac{2}{3}\mu \implies \zeta = \lambda + \frac{2}{3}\mu.$$

If thermodynamics is not involved, $\zeta = 0$, Eq. (1.27) is valid and $\lambda = -\frac{2}{3}\mu$.

In the light of the considerations above about the stress tensor $\boldsymbol{\sigma}$, the momentum Eq. (1.20) may be written as

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla\cdot(\rho\mathbf{u}\mathbf{u}^T) = \rho\mathbf{g} + \nabla\cdot\boldsymbol{\sigma} \overset{(1.23)}{=} \rho\mathbf{g} - \nabla p + \nabla\cdot\mathbf{d}. \tag{1.30}$$

For a Newtonian fluid, p and $\mathbf{d}$ verify Eq. (1.27) if thermodynamic is not involved, otherwise they respect Eq. (1.29) or (1.28).

1.1.5.4 Derivation of the energy conservation equation

The law for the conservation of energy (introduced in §1.1.2) states that the total amount of energy in a closed and isolated system remains constant over time. However, for the thermodynamic processes, the energy conservation law is adapted, and it translates into the *first law of thermodynamics*: in a closed system (such as a fluid parcel), the rate of change of the total energy of the system is equal to the sum of the rate of work done on the system due to the body and surface forces and the net flux of heat supplied to the system.

Considering the total energy $\mathcal{E}$ as the sum of the thermal energy and the kinetic energy, then the total energy possessed by a fluid parcel at time t is

$$\mathcal{E}(t) \coloneqq \int_{V_t} \rho e\, dV + \int_{V_t} \rho\frac{|\mathbf{u}|^2}{2} dV = \int_{V_t} E\, dV, \quad E \coloneqq \rho\left(e + \frac{|\mathbf{u}|^2}{2}\right), \tag{1.31}$$

where e is the specific internal energy, $|\mathbf{u}|^2/2$ is the specific kinetic energy, and E is the total energy density.

For the first principle of thermodynamics, the rate of change of the energy in the fluid parcel is equal to the sum of the rate of work done on the parcel and the heat received from the surrounding bodies, namely

$$\frac{d}{dt}\mathcal{E}(t) = \mathcal{L}(t) + \mathcal{Q}(t), \tag{1.32}$$

where the two terms on the right-hand side represent the work and heat contributions respectively.

Thanks to the Reynolds transport theorem II, the rate of change of energy expresses as follows

$$\begin{aligned}
\frac{d}{dt}\mathcal{E}(t) &\overset{(1.31)}{=} \frac{d}{dt}\int_{V_t} \rho\left(e + \frac{|\mathbf{u}|^2}{2}\right) dV \\
&\overset{(1.16)}{=} \int_{V_t} \rho\frac{D}{Dt}\left(e + \frac{|\mathbf{u}|^2}{2}\right) dV \\
&= \int_{V_t} \left[\frac{\partial E}{\partial t} + \nabla\cdot(E\mathbf{u})\right] dV,
\end{aligned} \tag{1.33}$$

where the last equation is obtained with computations similar to those present in Eq. (1.17):

$$\begin{aligned}
\rho\frac{D}{Dt}\left(e+\frac{|\mathbf{u}|^2}{2}\right) &= \rho\frac{\partial}{\partial t}\left(e+\frac{|\mathbf{u}|^2}{2}\right)+\rho\mathbf{u}\cdot\nabla\left(e+\frac{|\mathbf{u}|^2}{2}\right)\\
&= \rho\frac{\partial}{\partial t}\left(e+\frac{|\mathbf{u}|^2}{2}\right)+\nabla\cdot\left[\rho\mathbf{u}\left(e+\frac{|\mathbf{u}|^2}{2}\right)\right]-\left[\nabla\cdot(\rho\mathbf{u})\right]\left(e+\frac{|\mathbf{u}|^2}{2}\right)\\
&\overset{(1.13)}{=} \rho\frac{\partial}{\partial t}\left(e+\frac{|\mathbf{u}|^2}{2}\right)+\nabla\cdot\left[\rho\mathbf{u}\left(e+\frac{|\mathbf{u}|^2}{2}\right)\right]+\frac{\partial\rho}{\partial t}\left(e+\frac{|\mathbf{u}|^2}{2}\right)\\
&\overset{(1.31)}{=} \frac{\partial E}{\partial t}+\nabla\cdot(E\mathbf{u}).
\end{aligned}$$

The term $\mathcal{L}$ is related to the power of the external forces acting on the system. Reminding that the mechanical power of a fluid flow is the rate at which the work (exerted by the forces) is done, the power is the product of the bulk and contact forces exerted on the control volume and its velocity. Assuming that gravity is the only bulk force acting on the system and that the contact forces are entirely described by the stress tensor $\boldsymbol{\sigma}$ defined in Eq. (1.23), then $\mathcal{L}$ is expressed as:

$$\begin{aligned}
\mathcal{L}(t) &= \int_{V_t}\rho\mathbf{g}\cdot\mathbf{u}dV+\int_{\partial V_t}\mathbf{n}\cdot(\boldsymbol{\sigma}\mathbf{u})dS\\
&= \int_{V_t}\Big[\rho\mathbf{g}\cdot\mathbf{u}+\nabla\cdot(\boldsymbol{\sigma}\mathbf{u})\Big]dV\\
&\overset{(1.23)}{=} \int_{V_t}\Big[\rho\mathbf{g}\cdot\mathbf{u}-\nabla\cdot(p\mathbf{u})+\nabla\cdot(\mathbf{d}\mathbf{u})\Big]dV.
\end{aligned}\tag{1.34}$$

The $\mathcal{Q}$ term is related to two heat exchange phenomena: (i) the volumetric heating such as absorption or emission of radiation, and (ii) the heat transfer across the surface due to temperature gradients, i.e., thermal conduction. By naming $r = r(\mathbf{x}, t)$ the rate of volumetric heat addition per unit mass and considering the heat flux $\mathbf{q}_{cond} = \mathbf{q}_{cond}(\mathbf{x}, t)$ due to thermal conduction at the boundaries, the expression of $\mathcal{Q}$ is:

$$\mathcal{Q}(t) = \int_{V_t}\rho r dV-\int_{\partial V_t}\mathbf{q}_{cond}\cdot\mathbf{n}dS = \int_{V_t}\Big[\rho r-\nabla\cdot\mathbf{q}_{cond}\Big]dV,\tag{1.35}$$

in particular, the conductive heat flux depends on the temperature gradient as $\mathbf{q}_{cond} = -k\nabla T$, where k is the thermal conductivity and $T = T(\mathbf{x}, t)$ is the temperature.

By using what we derived in Eqs. (1.33), (1.34), and (1.35) to reassemble Eq. (1.32), we obtain the integral form of the energy conservation equation

$$\int_{V_t}\left\{\frac{\partial E}{\partial t}+\nabla\cdot(E\mathbf{u})-\Big[\rho\mathbf{g}\cdot\mathbf{u}-\nabla\cdot(p\mathbf{u})+\nabla\cdot(\mathbf{d}\mathbf{u})+\rho r-\nabla\cdot\mathbf{q}_{cond}\Big]\right\}dV = 0.$$

Since the previous condition must be verified for every fluid parcel V, the integrand must be null, so we find the differential expression of the energy conservation equation:

$$\frac{\partial E}{\partial t}+\nabla\cdot(E\mathbf{u}) = \rho\mathbf{g}\cdot\mathbf{u}-\nabla\cdot(p\mathbf{u})+\nabla\cdot(\mathbf{d}\mathbf{u})+\rho r-\nabla\cdot\mathbf{q}_{cond}.\tag{1.36}$$

1.1.5.5 Governing equations system, constitutive and state equations

The previous paragraphs showed the derivation of the Navier-Stokes governing equations for a viscous fluid in the presence of heat conduction and heat supply; we also assumed gravity to be the only external body force. We obtained the equations for the mass (1.13), momentum (1.30), and energy conservation (1.36) and we gather them together in the following system:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{1.37a}$$

$$\frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot \left(\rho \mathbf{u}\mathbf{u}^T\right) = \rho \mathbf{g} - \nabla p + \nabla \cdot \mathbf{d}, \tag{1.37b}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot (E\mathbf{u}) = \rho \mathbf{g} \cdot \mathbf{u} - \nabla \cdot (p\mathbf{u}) + \nabla \cdot (\mathbf{d}\mathbf{u}) + \rho r - \nabla \cdot \mathbf{q}_{cond} \tag{1.37c}$$

where we recall that $\rho(\mathbf{x}, t)$ is the fluid density (that might depend on other thermodynamic variables), $\mathbf{u}(\mathbf{x}, t)$ is the velocity vector field (with components $\mathbf{u} = (u, v, w)$), $\mathbf{g}$ is the gravity acceleration, $p(\mathbf{x}, t)$ is the pressure, $\mathbf{d}(\mathbf{x}, t)$ is the viscous stress tensor, $E(\mathbf{x}, t)$ is the density of total energy, $r(\mathbf{x}, t)$ is the rate of volumetric heat addition per unit mass, $\mathbf{q}_{cond}(\mathbf{x}, t)$ is the heat flux due to thermal conduction, dependent on the temperature gradient as $\mathbf{q}_{cond} = -k\nabla T$ where k is the thermal conductivity.

Usually, gravity $\mathbf{g}$ and heat supply r are prescribed. The viscous stress tensor $\mathbf{d}$ has the most generic expression defined in Eq. (1.23). In the case that the fluid exhibit a Newtonian behavior, the tensor $\mathbf{d}$ assumes the form reported in Eq. (1.26), which presents the viscosity coefficients μ and λ that represent two parameters of the problem. These two are always a property of the fluid and sometimes may depend on the unknowns of the problems by *constitutive* assigned equations. The total energy E is the sum of internal and kinetic energy $E = \rho e + \rho \frac{|\mathbf{u}|^2}{2}$, see Eq. (1.31), so it depends on density ρ, velocity $\mathbf{u}$, and internal specific energy e. This last unknown is a state variable, such as temperature and pressure; hence it depends on those variables through *state equations* that depend on the fluid. For example, in the case of an ideal gas, the internal energy is proportional to its mass, which is determined by the number of moles n, to its temperature T, and its molar heat capacity (at constant volume) c_V

$$e = c_V n T.$$

It remains to discuss the role of pressure p. To do that, we assume Newtonian behavior. If p is the thermodynamic pressure, then the deviatoric stress $\mathbf{d}$ follows Eq. (1.29) or (1.28) and there is a *state equation* that relates pressure with the other state variables. For instance, in the case of a perfect gas the following relation holds:

$$\rho = \frac{p\, m_g}{\mathcal{R} T},$$

where m_g is the gas molecular weight and $\mathcal{R}$ is the ideal gas constant. If the pressure is a mechanical quantity, the deviatoric stress $\mathbf{d}$ follows Eq. (1.27), and the pressure field p (apart from a constant value) is determined by the velocity distribution. To conclude, we assign the values of μ and λ with the constitutive equations and use the state equations for e and p, so the system of governing equations (1.37) has 5 equations (because the momentum Eq. (1.37b) contains 3 equations) and 5 unknowns ρ, $\mathbf{u}$, and T. Moreover, initial and boundary conditions need to be also assigned as we will see in §1.1.7.

The Navier-Stokes Eqs. (1.37) form a system of non-linear and coupled PDEs that is therefore very difficult to solve analytically. To date, there is no general closed-form solution for these equations. Before discussing the differential nature of the Navier-Stokes Eqs. (1.37), we recall that in the theory of scalar PDEs, according to a classic classification, it is common to distinguish them between:

- elliptic PDEs,
- parabolic PDEs,
- hyperbolic PDEs;

the first two types do not occur in the first-order case. *Elliptic* PDEs describe equilibrium problems such as, for example, the *Laplace equation* which describes the steady-state temperature distribution in the absence of heat sources. Conversely, *parabolic* and *hyperbolic* equations model evolutive problems, where the variables change not only in space but also in time. The *parabolic* PDEs have solutions characterized by irreversibility and the information travels in space at infinite speed (i.e. a perturbation prescribed at a fixed location in space immediately affect all the spatial domain). The main example of this type of equations is the *heat equation*. The *wave equation*, instead, is the most famous second order equation of the *hyperbolic* type; first order equations like the *transport equation* presented in §1.1.4 are another example of hyperbolic PDE.

Coming back to the Navier-Stokes system of Eqs. (1.37), we see that the equations show a mixed hyperbolic and parabolic nature. In fact, the continuity equation is purely hyperbolic. In contrast, the equations for the momentum and energy are parabolic, since the terms with the viscous stress tensor $\mathbf{d}$ and the term with the thermal conduction $\mathbf{q}_{cond}$ involve second-order spatial derivatives.

1.1.5.6 Incompressible Navier-Stokes equations

The Navier-Stokes Eqs. (1.37) model the motion of a generic Newtonian fluid. Additional hypotheses on the fluid physical properties or about the flow characteristics may simplify the equations. For the applications we are interested in, the hypothesis of incompressible flow is assumed and, because of it, the equations undergo important variations.

We pay attention to the terminology, and we distinguish between *incompressible fluids* and *incompressible flows*. In common sense, compressibility is a physical property of the fluids themselves, but the term incompressible may also be associated with a flow regime. Many texts that treat these concepts often do not distinguish between them because the results that these conditions produce are similar.

Definition 1.1. *An* incompressible fluid *is a fluid with constant density*

$$\rho(\mathbf{x}, t) = const. \tag{1.38}$$

Definition 1.2. *A* flow *is said* incompressible *(or isochoric) when the density of each fluid parcel that moves with the flow remains constant. This is equivalent to saying that the material derivative (defined in Eq. (1.4)) of density vanishes:*

$$\frac{D\rho}{Dt} = \frac{\partial \rho}{\partial t} + (\nabla \rho) \cdot \mathbf{u} = 0. \tag{1.39}$$

The introduction of the incompressible flow condition inside the mass conservation modifies it as follows

$$\begin{aligned}\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= \frac{\partial \rho}{\partial t} + (\nabla \rho) \cdot \mathbf{u} + \rho \, (\nabla \cdot \mathbf{u}) \\ &\overset{(1.4)}{=} \frac{D\rho}{Dt} + \rho \, (\nabla \cdot \mathbf{u}) \\ &\overset{(1.39)}{=} \rho \, (\nabla \cdot \mathbf{u}) \\ &\overset{(1.37a)}{=} 0,\end{aligned}$$

and, hence, the continuity equation under the incompressible flow condition becomes the kinematic constraint of divergence-free velocity field (for fluids with a nonzero density) instead of a dynamic equation:

$$\nabla \cdot \mathbf{u} = 0 \tag{1.40}$$

which corresponds to the condition of volumes conservation[1].

Notice that the incompressible flow condition, Eq. (1.39) allows the fluid to have a different density for each parcel, namely a non-homogeneous density distribution.

If one considers the case of an incompressible fluid, namely one restricts to the case of constant density, Eq. (1.38), this trivially implies that the partial derivatives of density are null; in such a situation, it descends that the incompressible flow condition of Eq. (1.39) is trivially verified, and therefore we find, again, that the mass conservation equation results to be the divergence-free velocity condition:

$$\rho(\mathbf{x}, t) = const \implies \begin{cases} \partial_t \rho = 0, \\ \nabla \rho = 0, \end{cases} \implies \nabla \cdot \mathbf{u} = 0. \tag{1.41}$$

It is not possible to say the opposite, namely that the incompressible flow condition implies a constant and homogeneous density field.

Even the momentum conservation Eq. (1.37b) changes its expression under the hypothesis of incompressible flow. We first focus on the hyperbolic terms to see how they change, then show the consequences of the incompressibility assumption on the stress tensor. Finally, expanding the hyperbolic terms by the chain rules, the density variable moves out of the differential terms, and it appears as a scalar that multiplies both the

[1] The volume of a fluid parcel V at time t is $\mathcal{V}_V(t) := \int_{V_t} dV$. The conservation of the volume of the fluid parcel means that $\dfrac{d\mathcal{V}_V(t)}{dt} = 0$, therefore by using the Reynolds transport Th.1.2 with $f \equiv 1$ we obtain

$$\frac{d\mathcal{V}_V(t)}{dt} = \frac{d}{dt} \int_{V_t} dV \overset{(1.12)}{=} \int_{V_t} \nabla \cdot \mathbf{u} \, dV = 0;$$

then the conservation of volume translates in the condition of null divergence $\nabla \cdot \mathbf{u}$.

transient term and the advective term:

$$\begin{aligned}\frac{\partial(\rho\mathbf{u})}{\partial t}+\nabla\cdot\left(\rho\mathbf{u}\mathbf{u}^T\right) &\overset{(1.43)}{=} \frac{\partial\rho}{\partial t}\mathbf{u}+\rho\frac{\partial\mathbf{u}}{\partial t}+\left[\nabla\cdot(\rho\mathbf{u})\right]\mathbf{u}+(\rho\mathbf{u}\cdot\nabla)\,\mathbf{u}\\ &= \underbrace{\left(\frac{\partial\rho}{\partial t}+\left[\nabla\cdot(\rho\mathbf{u})\right]\right)}_{=0 \text{ for Eq. (1.37a)}}\mathbf{u}+\rho\frac{\partial\mathbf{u}}{\partial t}+\rho\,(\mathbf{u}\cdot\nabla)\,\mathbf{u}\\ &= \rho\frac{\partial\mathbf{u}}{\partial t}+\rho\,(\mathbf{u}\cdot\nabla)\,\mathbf{u}+\underbrace{\rho(\nabla\cdot\mathbf{u})\mathbf{u}}_{=0 \text{ for Eq. (1.40)}}\\ &\overset{(1.44)}{=} \rho\left(\frac{\partial\mathbf{u}}{\partial t}+\nabla\cdot(\mathbf{u}\mathbf{u}^T)\right),\end{aligned} \tag{1.42}$$

where the first and last equalities use the following non trivial differential equivalences

$$\nabla\cdot\left(\rho\mathbf{u}\mathbf{u}^T\right)=\left[\nabla\cdot(\rho\mathbf{u})\right]\mathbf{u}+\left[\rho\,(\mathbf{u}\cdot\nabla)\right]\mathbf{u}, \tag{1.43}$$

$$\nabla\cdot\left(\mathbf{u}\mathbf{u}^T\right)=(\nabla\cdot\mathbf{u})\,\mathbf{u}+(\mathbf{u}\cdot\nabla)\,\mathbf{u}. \tag{1.44}$$

Since the second equation is a simpler version of the former one, only the first equation is proved (recalling that $\mathbf{u}$ has the components (u,v,w)):

$$\begin{aligned}\nabla\cdot(\rho\mathbf{u}\mathbf{u}^T) &= \nabla\cdot\begin{bmatrix}\rho u^2 & \rho uv & \rho uw\\ \rho vu & \rho v^2 & \rho vw\\ \rho wu & \rho wv & \rho w^2\end{bmatrix}=\begin{bmatrix}\partial_x(\rho u^2)+\partial_y(\rho uv)+\partial_z(\rho uw)\\ \partial_x(\rho vu)+\partial_y(\rho v^2)+\partial_z(\rho vw)\\ \partial_x(\rho wu)+\partial_y(\rho wv)+\partial_z(\rho w^2)\end{bmatrix}\\ &= \begin{bmatrix}\partial_x(\rho u)u+\partial_y(\rho v)u+\partial_z(\rho w)u\\ \partial_x(\rho u)v+\partial_y(\rho v)v+\partial_z(\rho w)v\\ \partial_x(\rho u)w+\partial_y(\rho v)w+\partial_z(\rho w)w\end{bmatrix}+\begin{bmatrix}(\rho u)\partial_x u+(\rho v)\partial_y u+(\rho w)\partial_z u\\ (\rho u)\partial_x v+(\rho v)\partial_y v+(\rho w)\partial_z v\\ (\rho u)\partial_x w+(\rho v)\partial_y w+(\rho w)\partial_z w\end{bmatrix}\\ &= (\partial_x(\rho u)+\partial_y(\rho v)+\partial_z(\rho w))\begin{bmatrix}u\\ v\\ w\end{bmatrix}+((\rho u)\partial_x+(\rho v)\partial_y+(\rho w)\partial_z)\begin{bmatrix}u\\ v\\ w\end{bmatrix}\\ &= \left[\nabla\cdot(\rho\mathbf{u})\right]\mathbf{u}+\left[\rho\mathbf{u}\cdot\nabla\right]\mathbf{u}.\end{aligned}$$

The incompressible flow condition may be applied to the stress tensor too (defined in Eq. (1.26)), and its expression simplifies and loses the term related with the damping associated with the volumetric straining:

$$\boldsymbol{\sigma}\overset{(1.26)}{=}-p\mathbf{I}+\underbrace{2\mu\mathbf{E}+\lambda(\nabla\cdot\mathbf{u})\mathbf{I}}_{\mathbf{d}}\overset{(1.40)}{=}-p\mathbf{I}+\underbrace{2\mu\mathbf{E}}_{\mathbf{d}}. \tag{1.45}$$

Notice also that the thermodynamic is not involved, because no volume variations are allowed in the incompressible condition, hence p is intended as the mechanical pressure, $p=p_{mech}$.

Under the incompressible flow hypothesis, the density "moves" out of the hyperbolic terms of the momentum equation, see Eq. (1.42). Therefore both the left and right sides of the momentum equation (1.37b) may be divided by ρ and one finds a modified version of the momentum equation

$$\frac{\partial\mathbf{u}}{\partial t}+\nabla\cdot(\mathbf{u}\mathbf{u}^T)=\mathbf{g}-\frac{\nabla p}{\rho}+\frac{1}{\rho}\nabla\cdot(2\mu\mathbf{E}). \tag{1.46}$$

Being mostly interested in modeling Newtonian fluids and incompressible flows, we rename the deviatoric stress tensor under these precise conditions as $\boldsymbol{\tau}$

$$\boldsymbol{\tau} := 2\mu\mathbf{E} \overset{(1.24)}{=} \mu\left[\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right], \tag{1.47}$$

and in the following we only refer to this notation.

For an incompressible fluid, namely in the case of constant density, Eq. (1.38), the momentum equation may further simplify its expression: ρ can move inside the pressure gradient and the divergence of the viscous stress, obtaining

$$\frac{\partial\mathbf{u}}{\partial t} + \nabla\cdot(\mathbf{u}\mathbf{u}^T) = \mathbf{g} - \nabla\frac{p}{\rho} + \nabla\cdot(2\nu\mathbf{E}), \tag{1.48}$$

where the coefficient ν is called *kinematic viscosity* and defined as

$$\nu := \frac{\mu}{\rho}.$$

In such simplified case of incompressible fluid, the Eq. (1.48) is accompanied only by the condition $\nabla\cdot\mathbf{u} = 0$.

Coming back to the case of a viscous Newtonian fluid, incompressible flow and with possibly non-constant and not uniform density, the governing equations are

$$\frac{\partial\rho}{\partial t} + \nabla\cdot(\rho\mathbf{u}) = 0, \tag{1.49a}$$

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla\cdot\left(\rho\mathbf{u}\mathbf{u}^T\right) = \rho\mathbf{g} - \nabla p + \nabla\cdot\boldsymbol{\tau}, \tag{1.49b}$$

namely a transport equation for the fluid density $\rho(\mathbf{x},t)$ and the momentum equation with the viscous stress tensor under the incompressible condition $\boldsymbol{\tau}(\mathbf{x},t)$. As observed, the pressure p is intended as the mechanical pressure, so it depends on the pressure distribution at the boundary. The system (1.49) is composed by 4 equations (because the momentum equation is written in vectorial formalism, but consists of 3 scalar equations) in 4 unknowns: ρ and $\mathbf{u} = (u,v,w)$. Even if some energy equation accompanies the system, there would not be a state equation (linking energy, pressure, and density) to add because thermodynamics is not involved. For example, in Chapter 2 we show that an equation for temperature may be added to the system, as a sort of energy equation, without accounting for any thermodynamic behavior. Also, such an equation would be coupled with the system only if a variable (such as ρ) or a parameter (like the dynamic viscosity μ) depends on temperature through a constitutive equation.

There is one last thing to notice about the term involving the divergence of the viscous stress tensor. Because of the definition of $\boldsymbol{\tau}$, it may split in two terms

$$\nabla\cdot\boldsymbol{\tau} \overset{(1.47)}{=} \nabla\cdot\left\{\mu\left[\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right]\right\} = \nabla\cdot(\mu\nabla\mathbf{u}) + \nabla\cdot[\mu(\nabla\mathbf{u})^T].$$

We compute the two resulting terms. The first one is

$$\nabla\cdot(\mu\nabla\mathbf{u}) = \nabla\cdot\left(\mu\begin{bmatrix}\partial_x u & \partial_y u & \partial_z u\\ \partial_x v & \partial_y v & \partial_z v\\ \partial_x w & \partial_y w & \partial_z w\end{bmatrix}\right) = \begin{bmatrix}\partial_x(\mu\partial_x u) + \partial_y(\mu\partial_x v) + \partial_z(\mu\partial_x w)\\ \partial_x(\mu\partial_y u) + \partial_y(\mu\partial_y v) + \partial_z(\mu\partial_y w)\\ \partial_x(\mu\partial_z u) + \partial_y(\mu\partial_z v) + \partial_z(\mu\partial_z w)\end{bmatrix}.$$

Note that the partial derivatives of μ appear; since they are equal in any direction (because the fluid is Newtonian and therefore isotropic) then the order of derivatives may change in every term:

$$\begin{bmatrix} \partial_x(\mu\partial_x u) + \partial_x(\mu\partial_y v) + \partial_x(\mu\partial_z w) \\ \partial_y(\mu\partial_x u) + \partial_y(\mu\partial_y v) + \partial_y(\mu\partial_z w) \\ \partial_z(\mu\partial_x u) + \partial_z(\mu\partial_y v) + \partial_z(\mu\partial_z w) \end{bmatrix} = \begin{bmatrix} \partial_x\left(\mu(\partial_x u + \partial_y v + \partial_z w)\right) \\ \partial_y\left(\mu(\partial_x u + \partial_y v + \partial_z w)\right) \\ \partial_z\left(\mu(\partial_x u + \partial_y v + \partial_z w)\right) \end{bmatrix} = \begin{bmatrix} \partial_x\left(\mu(\nabla\cdot\mathbf{u})\right) \\ \partial_y\left(\mu(\nabla\cdot\mathbf{u})\right) \\ \partial_z\left(\mu(\nabla\cdot\mathbf{u})\right) \end{bmatrix} \overset{(1.40)}{=} \mathbf{0}.$$

As a consequence, the first term is null for isotropic fluids. The second term is

$$\nabla\cdot[\mu(\nabla\mathbf{u})^T] = \nabla\cdot\left(\mu\begin{bmatrix} \partial_x u & \partial_x v & \partial_x w \\ \partial_y u & \partial_y v & \partial_y w \\ \partial_z u & \partial_z w & \partial_z w \end{bmatrix}\right) = \begin{bmatrix} \partial_x(\mu\partial_x u) + \partial_y(\mu\partial_y u) + \partial_z(\mu\partial_z u) \\ \partial_x(\mu\partial_x v) + \partial_y(\mu\partial_y v) + \partial_z(\mu\partial_z v) \\ \partial_x(\mu\partial_x w) + \partial_y(\mu\partial_y w) + \partial_z(\mu\partial_z w) \end{bmatrix}.$$

In the case where μ is constant, it exits from the derivatives and the second term becomes

$$\mu\begin{bmatrix} \partial_x\partial_x u + \partial_y\partial_y u + \partial_z\partial_z u \\ \partial_x\partial_x v + \partial_y\partial_y v + \partial_z\partial_z v \\ \partial_x\partial_x w + \partial_y\partial_y w + \partial_z\partial_z w \end{bmatrix} = \mu\begin{bmatrix} \Delta u \\ \Delta v \\ \Delta w \end{bmatrix} = \mu\Delta\mathbf{u}.$$

We conclude that when the fluid is Newtonian and isotropic and the flow is incompressible, the viscosity term in the momentum equation assumes the following expression

$$\nabla\cdot\boldsymbol{\tau} = \nabla\cdot(\mu\nabla\mathbf{u}) + \nabla\cdot(\mu(\nabla\mathbf{u})^T) = \nabla\cdot(\mu(\nabla\mathbf{u})^T).$$

In addition, if viscosity is constant then we obtain

$$\nabla\cdot\boldsymbol{\tau} = \mu\Delta\mathbf{u}. \tag{1.50}$$

1.1.5.7 Rheology models

Many fluids we deal with in everyday life are Newtonian, such as water, oil, air, alcohol, and glycerol, but just as many are not. For this reason, we present the generalization of the Newtonian models, investigating the *rheologycal* properties of other materials. *Rheology* is the branch of physics that studies the response of solids and liquids to the application of surface forces, the deformation and flow of the matter under the influence of applied stress. Eugene C. Bingham coined the term *rheology* for the first time in 1920, taking inspiration from the Greek aphorism *panta rhei* that means "everything flows", in particular r'eoz (*rheo-*) stands for "flow" and log'ia (*-logìa*) for "the study of". This study has applications in multiple fields and disciplines such as material science, engineering, biology, pharmaceutics, geophysics and food studies.

We remind that the Newtonian fluids in incompressible flow have the viscous stress tensor defined (in Eq. (1.47)) as

$$\boldsymbol{\tau} = \mu(2\mathbf{E}),$$

where μ denotes the dynamic viscosity. A generalized model is considered for those non-Newtonian fluids for which the 4th Newtonian assumption from 1–4 does not hold, that is, the linear dependence on velocity derivatives, Eq. (1.25), anyway the constitutive equation of the shear stress may still be formalized as the Newtonian way as follows

$$\boldsymbol{\tau} = \mu_{app}(2\mathbf{E}), \tag{1.51}$$

and the proportionality coefficient μ_{app} is named as *apparent viscosity*.

For fluids that respect the *power-law* relationship introduced in 1926 by Herschel and Bulkley to express the shear stress as a function of shear rate for both Newtonian and non-Newtonian fluids, see [251], the apparent viscosity depends on the shear rate according to the formula

$$\mu_{app} := \widetilde{\mu}|2\mathbf{E}|^{N-1} + \frac{\tau_0}{|2\mathbf{E}|}, \tag{1.52}$$

where $|\cdot|$ denotes the tensor norm (corresponding to the matrix Frobenius norm, [106, page 55]), $\widetilde{\mu}$ is the flow consistency index (also called power-law viscosity) and $N \in \mathbb{R}$ is the pseudoplastic constant that indicates the degree of non-linearity between the shear stress and the shear rate. All these coefficients may depend on other parameters such as temperature. τ_0 is the yield stress and is a threshold for the shear stress: if the shear stress magnitude is smaller than that, then there is no deformation, and therefore (1.52) does not hold whenever $|\boldsymbol{\tau}| < \tau_0$. The *Bingham fluids* are those characterized by $\tau_0 > 0$. Consider, for example, a fluid that moves with a one-dimensional shear flow along the x_1 axis, as the one represented in Figure 1.5d, so that the only nonzero velocity gradient is $\frac{\partial u_1}{\partial x_2}$. The shear rate tensor $2\mathbf{E}$ and its norm $|2\mathbf{E}|$ result in

$$2\mathbf{E} = \frac{\partial u_1}{\partial x_2}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad |2\mathbf{E}| = \sqrt{2}\left|\frac{\partial u_1}{\partial x_2}\right|.$$

The only nonzero entries of the shear stress tensor are τ_{12} and τ_{21} that, in addition, are equal because of the symmetry of the tensor. We write the expression of τ_{12}, considering the Eqs. (1.51), (1.52), applied to a Bingham fluid:

$$\begin{cases} \tau_{12} = \widetilde{\mu}\left(\sqrt{2}\left|\dfrac{\partial u_1}{\partial x_2}\right|\right)^{N-1}\dfrac{\partial u_1}{\partial x_2} \pm \dfrac{\tau_0}{\sqrt{2}}, \text{ if } |\boldsymbol{\tau}| > \tau_0, \\ \dfrac{\partial u_1}{\partial x_2} = 0, \text{ if } |\boldsymbol{\tau}| \leqslant \tau_0, \end{cases}$$

meaning that there is no shear flow ($\frac{\partial u_1}{\partial x_2} = 0$) if the shear stress exerted is lower than the yield stress τ_0.

Giving a glimpse to Figure 1.7, a brief overview of the different types of rheological models that can be described with the power-law model of Eq. (1.52) is discussed. The consequent behavior of the apparent viscosity, according with Eq. (1.52), is displayed in Figure 1.8. Newtonian fluids correspond to the case $\tau_0 = 0$, $N = 1$. The *pseudoplastic fluids*, or *shear thinning fluids*, have $n < 1$ then the shear rate increases faster than the shear stress; shampoos and ketchup belong to this family. On the opposite, the *dilatant fluids*, or *shear thickening fluids*, present the complementary behavior and $N > 1$; a mixture 1:2.5 of water and cornstarch is a common representative of this rheology because it behaves like water to small forces, but it acts as a solid and resists to the impact of violent forces. Toothpaste is a familiar example of *Bingham plastic* rheology, $\tau_0 > 0$ and $N = 1$; the toothpaste tube needs an initial squeeze to get the toothpaste out, however, squeezing it harder will not make it flow any easier (in fact, looking at Figure 1.8, viscosity stays over the specific threshold $\widetilde{\mu}$). The mayonnaise is a case of *Bingham pseudoplastic* rheology which presents $\tau_0 > 0$ and $N < 1$; mayonnaise flows out from a bottle as soon as it is squeezed, and the harder the bottle is squeezed, the thinner the mayonnaise comes out. A *Bingham dilatant* rheology model, $\tau_0 > 0$ and $N > 1$, is used in snow avalanches modeling [150].

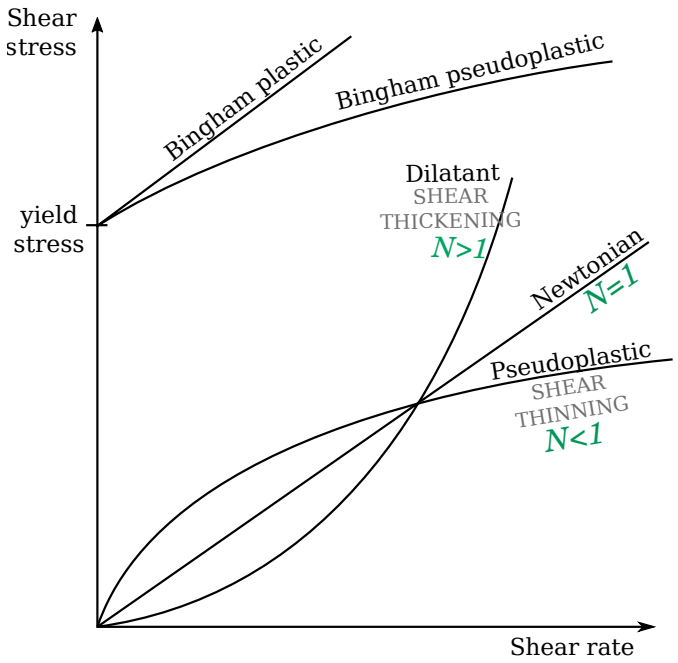


Figure 1.7: Classification of fluids according to the power law relation between the shear rate and the shear stress, see Eq. (1.52).

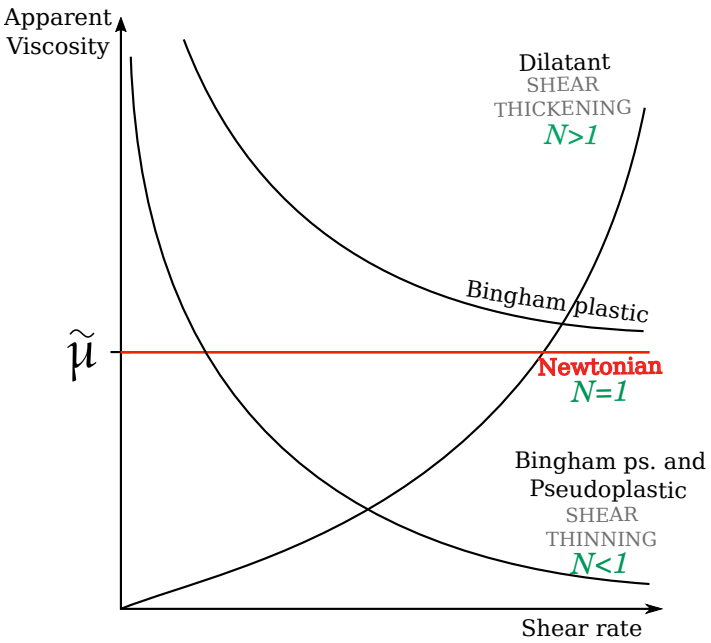


Figure 1.8: Apparent viscosity depending on the shear rate, see Eq. (1.52), of fluids that satisfy the power law. Note that the shear thickening plot represents a case where $N > 2$.

Viscosity is intrinsically related to the physical nature and chemical composition of involved materials, and it may also vary according to other factors related to the event dynamics. For example, it may depend on the concentration of sediments in geophysical flows involving erosion/deposition phenomena. Even the temperature may be accounted for as it happens, for example, in the Arrhenius model for liquids.

In this thesis, we are mainly interested in the specific application of lava flow modeling (we introduce it in the next §1.4.2), and fully molten lava exhibits Newtonian rheology (see Gonnermann and Manga [107]). However, the cooling of lava, crystallization process, and

degassing changes this behavior (see Pinkerton and Sparks [216], Cimarelli et al. [43], Lev et al. [165]) with the consequence that lava requires minimal shear stress to flow, therefore the Bingham model is the simplest approximation to it. Because of this motivation, in this thesis we will employ and refer to the Newtonian and Bingham plastic rheological models (also adopted in the VOLCFLOW software by Kelfoun and Vargas [145], for example). Furthermore, we plan to adopt also the Bingham pseudoplastic rheology in our model in future works.

1.1.5.8 Laminar and Turbulent flows

Some basic information about laminar and turbulent regimes is given here, without entering into the turbulence theory and modeling details (which represent research areas with great questions still to be explored), because it is far from our interest. For an introduction about turbulence, consult [9] and more specifically [255].

In everyday life experience, a river shows to the observer two scenarios: the flow may be calm as in Figure 1.9a or be impetuous and tumultuous as in Figure 1.9b when the water moves with swirls flowing downstream with great mixing.

(a) (b)

Figure 1.9: (a) Laminar flow in Arno river, Italy. (b) Turbulent flow in the middle Shire River, below one of the Nkula Falls hydroelectric dams, Malawi.

The different behavior is due to the relative magnitude of the viscosity forces and the inertial forces. In simple words, viscosity force could be considered a force that helps the fluid parcels to move with an order, one close to another. On the reverse, the greatest is the velocity, the more turbulent and chaotic the motion of fluid parcels might be. Thus, the way the fluid parcels move has consequences on the velocity field characteristics and leads to a classification of the flow:

- ***Laminar flow*** is a layer-like movement in which the fluid particles move in parallel directions, even if not necessarily at the same speeds.
- ***Turbulent flow*** is characterized by a chaotic distribution of particle motions; all the fluid parcels move in different directions and with curvilinear trajectories.

Consider the case of a viscous fluid moving horizontally in a laminar regime with a uniform velocity u_∞, see in Figure 1.10 the *green* arrows on the left that refer to the streamlines and that represent the uniform velocity profile. As soon as the fluid gets

in touch with a horizontal and stationary plate, the velocity profile changes because the friction with the plate constrains the velocity to decrease, see the *green* arrows in Figure 1.10 on the right. The *orange* dashed line approximately divides the free-stream region from the region where velocity differs from the value u_∞, such region is named *velocity boundary layer*, and we will deal again with this in §2.1.3. The *blue* arrows in Figure 1.10 within the plate and the dashed line represent the fluid parcels trajectories. In the beginning, the flow is laminar with the fluid parcel trajectories that follow the streamlines exactly. After a transition moment, if the velocity is large enough, the trajectories become curvilinear, with eddies of many sizes, and the flow turns into the turbulent regime. Despite that, there is still a region close to the plate where the viscous forces due to the friction with the bottom produce a laminar behavior.

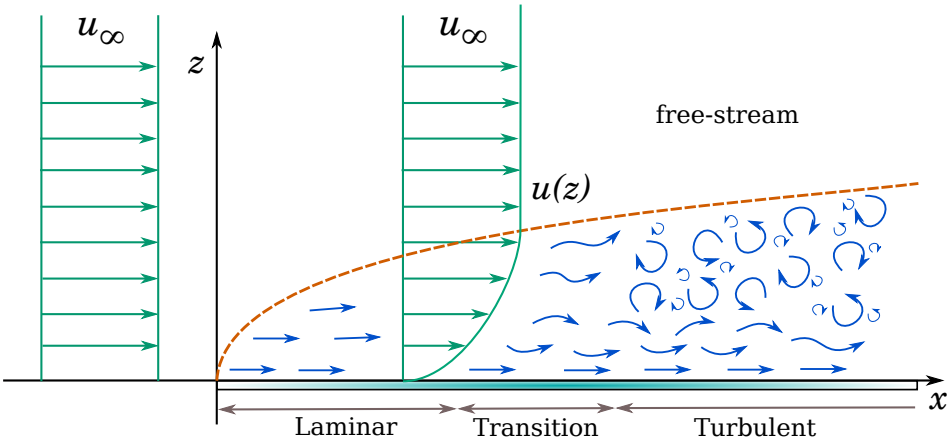


Figure 1.10: *Laminar and turbulent flow.* The straight and parallel *green* arrows refer to the streamlines and represent the velocity profile. The *blue* arrows symbolically represent the fluid parcels trajectories.

As anticipated, the relative magnitude of inertial force concerning viscous force rules the type of regime. Reynolds [224] introduced a dimensionless number, named *Reynolds number* and denoted as Re, which is proportional to the ratio between the inertial forces (caused by velocity) and the viscous forces. It helps to classify the type of regime. We show its derivation for a Newtonian fluid with constant viscosity in the incompressible flow condition. We consider the momentum Eq. (1.46) and use the result obtained for the viscosity term under such hypothesis; namely, we use Eq. (1.50), and we get the following expression for the momentum equation

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}^T) = \mathbf{g} - \frac{\nabla p}{\rho} + \frac{\mu}{\rho}\nabla^2 \mathbf{u}.$$

We rewrite the left-hand side, express that in terms of the material derivative (see Eq. (1.4)) and use the incompressible condition $\nabla \cdot \mathbf{u} = 0$:

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}^T) \overset{(1.44)}{=} \frac{\partial \mathbf{u}}{\partial t} + (\nabla \cdot \mathbf{u})\,\mathbf{u} + (\mathbf{u} \cdot \nabla)\,\mathbf{u} = \frac{D\mathbf{u}}{Dt}.$$

Adopting this expression, the momentum equation writes as follows

$$\frac{D\mathbf{u}}{Dt} = \mathbf{g} - \frac{\nabla p}{\rho} + \nu \nabla^2 \mathbf{u}, \tag{1.53}$$

where we also introduced the kinematic viscosity $\nu = \mu/\rho$. Consider some characteristic values (i.e., reference values depending on the problem): a characteristic length L_0, a characteristic velocity V_0, and density scale ρ_0. For example, when considering an open channel flow like a river, the depth-averaged velocity and the fluid depth are chosen as characteristic values V_0 and L_0; in other situations, the choice of the parameters changes. We define the dimensionless fluid variables:

$$\hat{\mathbf{u}} = \frac{\mathbf{u}}{V_0}, \quad \hat{\mathbf{x}} = \frac{\mathbf{x}}{L_0}, \quad \hat{p} = \frac{p}{\rho_0 V_0^2}.$$

In a similar fashion, we define a dimensionless time, a dimensionless density, a dimensionless spatial differential operator, and a dimensionless gravity acceleration:

$$\hat{t} = \frac{t}{L_0/V_0}, \quad \hat{\rho} = \frac{\rho}{\rho_0}, \quad \hat{\mathbf{g}} = \frac{L_0}{V_0^2}\mathbf{g}, \quad \hat{\nabla} = L_0\nabla.$$

We use these definitions to write the momentum equation in terms of the adimensional quantities

$$\frac{V_0^2}{L_0}\frac{D\hat{\mathbf{u}}}{D\hat{t}} = \frac{V_0^2}{L_0}\hat{\mathbf{g}} - \frac{\rho_0 V_0^2}{\rho_0 L_0}\frac{\hat{\nabla}\hat{p}}{\hat{\rho}} + \frac{\nu V_0}{L_0^2}\hat{\nabla}^2\hat{\mathbf{u}},$$

and after simplifying the coefficients, we obtain the following dimensionless equation

$$\frac{D\hat{\mathbf{u}}}{D\hat{t}} = \hat{\mathbf{g}} - \frac{\hat{\nabla}\hat{p}}{\hat{\rho}} + \frac{\nu}{L_0 V_0}\hat{\nabla}^2\hat{\mathbf{u}}.$$

The interesting thing in this formulation is the coefficient of the viscous term. It is the inverse of the *Reynolds number*, one of the most important dimensionless numbers in fluid dynamics:

$$\mathrm{Re} = \frac{V_0 L_0}{\nu}. \tag{1.54}$$

Problems characterized by similar Reynolds number are expected to exhibit similar fluid behavior. Intuitively, we could say that the Reynolds number measures the importance of inertia with respect to viscosity in Eq. (1.53)

$$\mathrm{Re} \approx \frac{D\mathbf{u}/Dt}{\nu\nabla^2\mathbf{u}}.$$

At low Reynolds numbers, the flow is mostly dominated by the viscous effects, which act as momentum diffusion, resulting in a laminar behavior, that is the case of Figure 1.9a. Whereas, at high Reynolds numbers, the flow tends to be turbulent because the inertia is dominant, as the situation of Figure 1.9b.

1.1.6 Systems of hyperbolic PDEs

Generally, in fluid dynamics, the solution of a system of PDEs is required, such as the Navier-Stokes system of Eqs. (1.37) introduced in §1.1.5 or the systems (2.40) and (2.59) derived in Chapter 2. Even though the system of PDEs we have to solve show a mixed differential nature, both parabolic and hyperbolic, we focus and investigate their hyperbolic setting. This choice is because we will use numerical schemes for PDEs that first solve the hyperbolic part of the equations and then the non-hyperbolic terms.

In the case of a system of transport equations (by focusing on the hyperbolic parts of the equations), the vector that gathers the conservative variables is defined:

$$\mathbf{q} = \begin{bmatrix} q_1 \\ \vdots \\ q_N \end{bmatrix},$$

and the vector of the fluxes is $\mathbf{f} = \left(\mathbf{f}^{(1)}, \mathbf{f}^{(2)}, \mathbf{f}^{(3)}\right)$ with the components in the three directions (if we consider the most generic case of 3 dimensional problem). The system is expressed by the following equation:

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{q}) = \mathbf{S}(\mathbf{q}) \quad \Longleftrightarrow \quad \frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{f}^{(1)}(\mathbf{q})}{\partial x} + \frac{\partial \mathbf{f}^{(2)}(\mathbf{q})}{\partial y} + \frac{\partial \mathbf{f}^{(3)}(\mathbf{q})}{\partial z} = \mathbf{S}(\mathbf{q}), \tag{1.55}$$

where $\mathbf{S}(\mathbf{q})$ is the vector of the source terms and possibly parabolic terms. We underline that, even though it is not explicitly written, the conservative quantities depend on the spatial coordinates $\mathbf{x}$ and time t. Thus, even if the single equations that constitute the system (1.55) are hyperbolic, the whole system might not be classified as *hyperbolic system*. In the following, we present the condition the system has to respect for being classified as hyperbolic.

The system of Eqs. (1.55) is *non-linear* in most of the cases, and knowing its *quasi-linear* expression might be useful. By applying the chain rule to the spatial derivative terms, the quasi-linear formulation of the system descends:

$$\frac{\partial \mathbf{q}}{\partial t} + \frac{d\mathbf{f}^{(1)}}{d\mathbf{q}}\frac{\partial \mathbf{q}}{\partial x} + \frac{d\mathbf{f}^{(2)}}{d\mathbf{q}}\frac{\partial \mathbf{q}}{\partial y} + \frac{d\mathbf{f}^{(3)}}{d\mathbf{q}}\frac{\partial \mathbf{q}}{\partial z} = \mathbf{S}(\mathbf{q}), \tag{1.56}$$

where the matrices $\dfrac{d\mathbf{f}^{(i)}}{d\mathbf{q}}$ are the Jacobian of the flux functions $\mathbf{f}^{(i)}(\mathbf{q})$.

Definition 1.3. *The quasi-linear system of PDEs (1.56) is said hyperbolic in a "point"* $(\mathbf{q}, \mathbf{x}, t)$ *if all the matrices* $\dfrac{d\mathbf{f}^{(i)}}{d\mathbf{q}}, i = 1, \ldots, 3$, *are diagonalizable with real eigenvalues, which means that in such a point the system verifies the* hyperbolic condition for systems.

Definition 1.4. *The nonlinear system of PDEs (1.55) is said hyperbolic in a point if its quasi-linear form (1.56) is hyperbolic in a point.*

The eigenvalues of the matrices are related with the velocity of propagation, playing the role of $\bar{u}$ in the simplest linear transport equation $\partial_t q + \bar{u}\partial_x q = 0$. Remember that there are some classes of matrices for which the system is certainly hyperbolic, for example, for symmetric matrices because they are always diagonalizable with distinct eigenvalues. Generally, we denote the eigenvalues of the three Jacobian matrices as:

$$\lambda_1^{(i)} \leqslant \cdots \leqslant \lambda_N^{(i)}, \quad i = 1, \ldots, 3. \tag{1.57}$$

Note that if the flux explicitly depends also on $\mathbf{x}$ and t, namely $\mathbf{f} = \mathbf{f}(\mathbf{q}, \mathbf{x}, t)$, we need a different treatment.

1.1.7 Initial and boundary conditions

Any system of PDEs must be associated with additional conditions in order to make the problem well-posed. Generally, the initial conditions are necessary for at least every variable that evolves in time. For the classic Navier-Stokes Eqs. (1.37), one may give the initial conditions for the variables $\rho, \mathbf{u}, E$:

$$\rho(\mathbf{x}, 0), \quad \mathbf{u}(\mathbf{x}, 0), \quad E(\mathbf{x}, 0).$$

The classic boundary conditions to use on a single variable are the Dirichlet and Neumann conditions, which supply a fixed value constraint on the variable itself or its gradient, respectively. Other boundary conditions are derived from different types of combinations of these two. Having a system of PDEs, the boundary conditions imposed on each variable must not conflict with the others. Obviously, the boundary conditions must not be in contradiction with the initial conditions. A common boundary condition in the presence of a *wall* is that the velocity component normal to the wall should be zero, which corresponds to a solid obstacle that fluid cannot pass through. In the case of a viscous fluid, there is another physical boundary condition one might wish to impose at a solid wall: the *no-slip* boundary condition, that produces a perfect contact, but with no-slip, between the fluid and the solid surface that constitutes the boundary. The no-slip boundary condition states that the tangential velocity component too should vanish at the wall along with the normal velocity so that fluid adjacent to the wall is stationary (this corresponds to a Dirichlet condition for velocity). This condition is expected due to friction between the wall and fluid molecules, which prevents molecules from slipping freely along the wall. However, this friction is present only in viscous fluids.

In hyperbolic problems, moreover, the boundary conditions must consider the eigenvalues of the Jacobian matrices of the fluxes; this is because the eigenvalues are associated with the solution propagation velocity. So, considering, for example, a flux that enters from the left side of the domain, propagates to the right, and exits on the right side of the domain, the direction of the flow determines the boundary conditions to impose: an input condition is necessary on the left whereas none condition is necessary on the right.

1.2 Finite Volume Method for hyperbolic problems

In the past, scientists had to rely on analytical skills to solve significant problems in mathematics, and thus they had to undergo rigorous training. Nevertheless, analytical solution methods are limited to highly simplified problems over simple geometries. When attempting to get an analytical solution to a physical problem, there is often the tendency to oversimplify the problem to make the mathematical model simple enough to warrant an analytical solution. Because of this tendency, one ends to obtain the *exact solution of an oversimplified model*.

Today's scientists have access to an enormous amount of computational power, allowing them to find *approximate solutions of less simplified models*. A mathematical model intended for a numerical solution is likely to represent the actual problem better. The approximation obtained by the numerical models consists of replacing the original terms of the equations with a set of expressions that a computer can solve.

When we plan to solve a mathematical model by a computer searching for a ***numerical solution***, we have to consider that computers have a finite amount of memory, whereby we have to select a set of discrete locations in space and/or time where the numerical solution

is computed. The discrete locations at which the variables are evaluated are defined by the ***numerical grid*** which is essentially a discrete representation of the geometric domain on which the problem has to be solved.

There are mainly three families of methods to solve the partial differential equations:

- **Finite difference methods:** the domain is discretized as numerical grid points, the numerical solution is a point-wise approximation at the points of the numerical grid, and the derivatives of the solution are approximated by finite differences, requiring a strong hypothesis on the regularity of the solution.

- **Finite element methods:** the domain is discretized to create a mesh composed of small parts (the *finite elements*). On each element, the solution is approximated by a linear combination of functions called *shape functions* and the coefficients of the linear combination are the unknowns of the algebraic problem obtained from the discretization. Then, variational methods are used to approximate the solution by minimizing an associated error function. The finite element method (FEM) belongs to the class of Galerkin methods whose starting point is the weak formulation of the differential problem, based on the concept of derivative in the sense of distributions. The FEM differs from other Galerkin methods in the local choice of polynomial shape functions, whereby a piece-wise polynomial function approximates the whole solution. Thus, the FEM is advantageous for complex geometries but has difficulties dealing with shocks and other evolving discontinuities.

- **Finite volume methods:** the domain is discretized into a finite number of cells (the *finite volumes*) and the numerical solution is approximated by the cell average, which is the integral of the variable over the grid cell divided by the volume of the cell. This family was born to solve problems with governing equations expressed as the integral form of a conservation law, such as the continuity equation for the conservation of the mass Eq. (1.49a), namely for

$$\int_{\tau}^{t+\Delta\tau} \int_{V} \left[\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) \right] dV dt = 0.$$

  Since the equations in integral form do not require the hypothesis of smoothness or regularity of the functions involved, the *finite volume methods* (FVM) do not impose restrictions and are suitable to capture discontinuities that are typical in fluid dynamics (as, for example, at the flow front).

Because of all the good properties concerning the *finite volume methods*, we use methods of this class in our work.

In this section, we give the fundamental knowledge about the FVM applied to hyperbolic problems; for a wider treatment of this topic, the reader can consult LeVeque [168].

We start describing the application of the FVM to a scalar equation in §1.2.1. Then we discuss some good convergence properties that the schemes belonging to the FVM family need, with a focus on stability, in §1.2.2. After, from §1.2.3 to §1.2.8, we present some relevant examples of numerical schemes and discuss their performances with particular attention to the discontinuity treatment. Lastly, in §1.1.6, we introduce the discretization strategy of the FVM for a system of PDEs in a multidimensional context shortly; we write

an example of a numerical scheme applied to a system, and then we comment on stability. Our presentation is an introduction to the FVM and sets the basis to understand our numerical scheme for the shallow-water equations described in chapter §3. Moreover, in the next section §1.3, we show the way the FVM is used to solve a generic PDE in the OpenFOAM context, and such numerical setting sets the basis for the numerical scheme for the 3D model described in chapter §4.

1.2.1 Scalar equation

For the sake of simplicity, we consider a problem with one scalar equation in one space dimension. The interval $[a, b]$ is the spatial domain and we use a uniform discretization dividing it into m, $m \in \mathbb{N}$, sub-intervals of amplitude $\Delta x := \frac{b-a}{m}$. We denote by C_i, for $i = 1, \dots, m$, the i-th sub-interval, and we refer to it as a *cell*, a *finite volume* or even a *control volume*. The interface between two cells C_i, C_{i+1} is located in position $x_{i+1/2}$, in particular

$$x_{i+\frac{1}{2}} = a + i\Delta x, \quad C_i = \left(x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}\right), \quad i = 1, \dots, m,$$

because we are using a uniform grid. The middle point of the cell C_i is said x_i. Figure 1.11 shows the domain discretization.

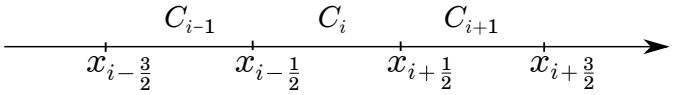


Figure 1.11: Example of spatial discretization.

We consider the unknown variable $q(x, t)$, $x \in [a, b], t \geqslant 0$, a conservative quantity that verifies the following integral form of the conservation equation

$$\int_\tau^{\tau+\Delta t} \int_\alpha^\beta \left[\frac{\partial q}{\partial t} + \frac{\partial f(q)}{\partial x}\right] dxdt = 0, \tag{1.58}$$

for every time $\tau \geqslant 0$, for every interval $[\alpha, \beta] \subseteq [a, b]$. By rearranging the terms, we get an equivalent expression of the equation:

$$\begin{aligned} \int_\tau^{\tau+\Delta t} \left[\frac{\partial}{\partial t} \int_\alpha^\beta q(x,t)dx\right] dt &= -\int_\tau^{\tau+\Delta t} \Big[f(q)|_\beta - f(q)|_\alpha\Big]dt, \\ \int_\alpha^\beta q(x,\tau+\Delta t)dx - \int_\alpha^\beta q(x,\tau)dx &= -\int_\tau^{\tau+\Delta t} \Big[f(q)|_\beta - f(q)|_\alpha\Big]dt, \end{aligned} \tag{1.59}$$

where the notation $f(q)|_\beta$ and $f(q)|_\alpha$ means the evaluation of the flux function at $x = \beta$ and $x = \alpha$.

From the discretizations of time and space descends, it that the Eq. (1.58) must be verified for every cell C_i and for every time interval $[t_n, t_{n+1}]$, then the discretized form of Eq. (1.58) results to be

$$\int_{C_i} q(x, t_{n+1})dx - \int_{C_i} q(x, t_n)dx = -\int_{t_n}^{t_{n+1}} \Big[f(q)|_{x_{i+\frac{1}{2}}} - f(q)|_{x_{i-\frac{1}{2}}}\Big]dt. \tag{1.60}$$

We indicate by Q_i^n the approximation of the integral average of the solution $q(x, t)$ over the cell C_i at the time step t_n:

$$Q_i^n \approx \frac{1}{\Delta x} \int_{C_i} q(x, t_n)dx. \tag{1.61}$$

According with this definition, we have that the left-hand side of Eq. (1.60) is approximated as $(Q_i^{n+1} - Q_i^n)\Delta x$. We need to determine an approximation also for the right-hand side terms.

1.2.1.1 Numerical flux

The approximation of the integral average, between t_n and t_{n+1}, of the real flux $f(q(x,t))$ at the interface $x_{i+\frac{1}{2}}$ is named ***numerical flux***, denoted as $F_{i+\frac{1}{2}}$:

$$F_{i+\frac{1}{2}} \approx \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} f\big(q\big(x_{i+\frac{1}{2}},t\big)\big)dt. \tag{1.62}$$

By using the definitions of numerical solution, Eq. (1.61), and of the numerical flux, Eq. (1.62), the exact integral Eq. (1.60) is approximated as follows:

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}\left(F_{i+\frac{1}{2}} - F_{i-\frac{1}{2}}\right). \tag{1.63}$$

A scheme that uses these approximations is said to be a ***conservative scheme*** since it guarantees that the integral of the numerical solution over the whole domain is preserved in the proper way, which means that the numerical approximation verifies Eq. (1.59) with $[\alpha,\beta] = [a,b]$. In fact, notice that the sum of the flux differences cancels out at every interface except for the fluxes at the extreme edges, as we can see by the following passages

$$\begin{aligned} Q_1^{n+1} + \cdots + Q_m^{n+1} = \\ = Q_1^n + \cdots + Q_m^n - \frac{\Delta t}{\Delta x}\left(F_{1+\frac{1}{2}} - F_{1-\frac{1}{2}} + \cdots + F_{m+\frac{1}{2}} - F_{m-\frac{1}{2}}\right) = \\ = Q_1^n + \cdots + Q_m^n - \frac{\Delta t}{\Delta x}\left(F_{m+\frac{1}{2}} - F_{1-\frac{1}{2}}\right). \end{aligned}$$

The numerical flux must be defined as a function of the numerical solution, i.e. as a function of the values $Q_i, i = 1,\ldots,m$, if we want an explicit treatment. Since we know that in hyperbolic problems the information travels at finite speed, we reasonably assume that, for small values of the time step, the flux $F_{i+\frac{1}{2}}$ is based only on Q_i^n and Q_{i+1}^n:

$$F_{i+\frac{1}{2}} = \mathcal{F}(Q_i^n, Q_{i+1}^n), \tag{1.64}$$

thus a definition of the numerical flux is explicit in time, but we will see in §1.2.8 that even the implicit treatment of the numerical flux is possible. From now on, we consider that the superscript of the numerical flux corresponds to the superscript of the temporal evaluation of the numerical solution. The FVM results in the following scheme:

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}\left[\mathcal{F}(Q_i^n, Q_{i+1}^n) - \mathcal{F}(Q_{i-1}^n, Q_i^n)\right], \tag{1.65}$$

that rules the time marching of the numerical solution. The scheme is completely defined once the numerical flux $F_{i+\frac{1}{2}}^n$ of Eq. (1.64) is specified: according with the possible definitions of the function $\mathcal{F}$, different methods descend.

Later in this section, from §1.2.3 to §1.2.8, we introduce various ways to define the numerical flux function leading to different FVM schemes.

The first type of methods we introduce (§1.2.3) are the centered schemes, for which the numerical flux is defined symmetrically with respect to the interface, and in this context we present two examples. In §1.2.3 we show the simplest definition for a symmetric numerical flux, namely the arithmetic mean of the fluxes across the interface; such simplicity has the drawback that produces unstable results over the discontinuities of the solution, even though the CFL condition (the necessary condition for stability, see §1.2.2.2) is respected. The second centered scheme presented (§1.2.3) is the Lax-Friedrichs scheme. Its definition is similar to the previous scheme but with a corrective diffusive term that prevents instabilities to occur. The defect of the scheme is related to the corrective term, for which the numerical results present too much diffusion and the discontinuities are smoothed.

The second type of method we introduce (§1.2.4) is the upwind of the simplest type, whose background idea is to use the knowledge of the flux velocity direction for the definition of the numerical flux. However, even though the upwind schemes are stable, they still present a diffusive behavior. After discussing these three first-order accurate schemes, we present the second-order Lax-Wendroff method (§1.2.5), which is more accurate and less dissipating than the previous ones. In particular, the Lax-Wendroff scheme produces much better numerical approximations of the smooth parts of the solution. However, together with these good features, the Lax-Wendroff scheme introduces instabilities in correspondence to the discontinuities in the form of oscillations; therefore, it may not be satisfactory. The four schemes described so far are *linear methods* but, in Theorem 1.5, we discover that the linear schemes that do not generate oscillations (namely that are stable under the CFL condition) are at most first-order accurate. Therefore, a higher-order method (allowing us to improve the accuracy) that does not generate oscillations cannot be linear, so we pass to examine non-linear higher-order schemes. To obtain such kinds of schemes, in §1.2.6 we introduce the concept of linear reconstruction for the numerical solution over each cell and use the related information about the solution at the interfaces to get better estimations of the numerical flux. In the context of the higher-order schemes with linear reconstructions, we dig into the oscillations generation for the advective equation with constant velocity (§1.2.6), we find that the *Total variation diminishing* (TVD) is the additional stability property that the higher-order schemes must respect to prevent oscillations (§1.2.6) and we introduce the *geometric limiters* (or *slope limiters*) as the tool to ensure the property (§1.2.6). In §1.2.7, we present an alternative point of view for the creation of higher-order TVD schemes offered by the *flux limiters*; however, it is possible to translate the slope limiters into flux limiters. Finally, in §1.2.8, we present the Kurganov-Noelle-Petrova scheme, which is an example of a higher-order TVD scheme. This method brings the advantage of better accuracy because it estimates, for each interface, the specific speed of propagation of discontinuities (accounting for a non-symmetric behavior at the interface). Moreover, this scheme may be applied to linear and non-linear hyperbolic equations, and it admits a semi-discrete expression.

1.2.2 Convergence

When solving a PDE with a numerical approach, knowing the accuracy and convergence properties of the methods adopted is important to ensure the numerical solution to be a sufficiently good approximation of the exact solution. Usually, the equations that model real problems do not have an exact solution to compare with the numerical solution; hence one must rely on a combination of the following techniques in order to understand better

the performances of the numerical schemes:

- *Validation on test problems.* The method is tested on a simple problem for which the exact solution is known or on problems for which a highly accurate comparison solution may be computed by other means. In some cases, experimental results may also be available for comparison.

- *Theoretical analysis of convergence and accuracy.* Prove that the method used converges to the correct solution as the grid is refined and obtain reasonable estimates of the numerical error that will be observed on any particular finite grid.

Here we present the theoretical analysis, but in our work, we validate the numerical methods also by using benchmarks related to our applications, see §3.4 and §4.7.

In order to talk about *accuracy* or *convergence*, quantifying the error done when approximating a function of space and time is necessary, and there are several ways to measure it. In Eq. (1.61), we defined Q_i^n as the approximation of the averaged value over a cell of the exact solution at time t_n. For comparison with the exact solution, we need to introduce q_i^n that represents the exact value that Q_i^n should approximate, namely

$$q_i^n := \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} q(\xi, t_n) d\xi.$$

In order to discuss the convergence, one picks a finite time interval $[0, T]$ over which we compute the approximated solution. Errors generally grow with time and with the number of time steps. Therefore, it would be unreasonable to expect that any finite grid would be capable of yielding good solutions at an arbitrarily large time $T \gg 0$ requiring a great number of time steps.

From now on, in this discussion about convergence, we use the superscript N to refer to the last time step, $T = N\Delta t$.

Definition 1.5 (Global error)**.** *The* global error *at the time $T = N\Delta t$ is*

$$\mathbf{E}^N = \mathbf{Q}^N - \mathbf{q}^N,$$

where $\mathbf{Q}^N = \left[Q_i^N\right]_i$ and $\mathbf{q}^N = \left[q_i^N\right]_i$ are vectors containing the approximated and exact averages of the solution $q(x,t)$ over the cells at the ending time.

A convergent method preserves this grid-function $\mathbf{E}^N$ bounded as the grid is refined. Assume that the ratio $\Delta t/\Delta x$ is fixed. Therefore, when Δt goes to 0, also the grid is refined, and we can speak about convergence with order s if the errors vanish like $\mathcal{O}(\Delta t^s)$ or as $\mathcal{O}(\Delta x^s)$ which are the same thing.

To quantify the error at a fixed time, a norm is chosen among the standard set of p-norms that are commonly used:

$$\|\mathbf{E}\|_p = \left(\Delta x \sum_{i=-\infty}^{\infty} |E_i|^p \right)^{1/p}, \tag{1.66}$$

that are the discrete analogs of the function-space norms:

$$\|E\|_p = \left(\int_{-\infty}^{\infty} |E(x)|^p dx \right)^{1/p}. \tag{1.67}$$

The presence of the factor Δx in Eq. (1.66) ensures the correct scaling and order of accuracy as the grid is refined. We highlight that in the framework of the conservation laws, the 1-norm (namely $p = 1$) is commonly adopted because the integrals of the solution itself are of particular importance. Anyway, we do not specify the norm and use the generic notation $\|\cdot\|$.

Definition 1.6 (Convergent method and order of convergence)**.** *A numerical method for PDE is said to be* convergent *at time T with respect to the norm $\|\cdot\|$ if:*

$$\lim_{\substack{\Delta t \to 0 \\ N\Delta t = T}} \left\|\mathbf{E}^N\right\| = 0.$$

A numerical method for PDE is said to be accurate of order s *if:*

$$\left\|\mathbf{E}^N\right\| = \mathcal{O}(\Delta t^s), \quad as \quad \Delta t \to 0.$$

Since it is generally impossible to determine a closed-form expression for the global error after so many time steps, another approach is usually adopted, which consists of studying two other features of the problem:

- *Consistency*: the error introduced in a single time step is small, showing that the method is consistent with the differential equation;
- *Stability*: the local error do not grow dramatically and a bound for the global error can be expressed in terms of the local one.

1.2.2.1 Consistency

An explicit numerical method may be written as follows

$$\mathbf{Q}^{n+1} = \mathcal{N}(\mathbf{Q}^n),$$

where $\mathcal{N}(\cdot)$ denotes the numerical operator that maps the solution (possibly approximated) at a certain time step to the solution approximation at the next time step. The error committed using the numerical method at a single time step is the difference between the result of the numerical operator applied to the true (cell-averaged) solution at some time $\mathcal{N}(\mathbf{q}^n)$ and the true solution at the next time $\mathbf{q}^{n+1}$.

Definition 1.7 (Local truncation error)**.** *The* local truncation error *is defined by dividing the one-step error by Δt:*

$$\boldsymbol{\tau}^n = \frac{\mathcal{N}(\mathbf{q}^n) - \mathbf{q}^{n+1}}{\Delta t}$$

Moreover it is quite easy to investigate the local truncation error in the case of smooth solutions, because it is well approximated by simple Taylor series expansions.

Definition 1.8 (Consistent method)**.** *The numerical method for PDE is said* consistent *with the differential equation if for any n and for all smooth functions $q(x,t)$, satisfying the differential equation*

$$\boldsymbol{\tau}^n \longrightarrow 0 \quad as \quad \Delta t \to 0.$$

For an exhaustive discussion about consistency see [168, chapter 8].

1.2.2.2 Stability and CFL condition

By using a conservative finite volume method based on the integral form of the conservation laws, we hope to get a numerical solution that correctly approximates the weak solutions of the conservation laws. Lax and Wendroff [162] proved this in the following theorem (not formally written):

Theorem 1.4 (Lax-Wendroff theorem)**.** *For a conservative and consistent method that applies to scalar equations (or even to non-linear system of conservation laws), in the case a sequence of numerical approximations converges in an appropriate sense (see [162],[168, chapter 12, page 240]) to a function $q(x,t)$ as the grid refines, then the limit function is a weak solution of the conservation law.*

Thanks to what was stated, we can trust the solution we compute. Even though we usually do not compute an entire sequence, if the numerical solutions look reasonable and the discontinuities are well-treated, we can believe in having good approximations to some weak solutions.

Despite the great relevance of this theorem, it does not provide any results about convergence. Also, convergence requires some form of stability, as we see in the following. Although we do not give a detailed study of stability analysis, we provide the basic ideas of stability theory. (*i*) We start introducing the *CFL Condition* which is *necessary* for a finite volume method that is expected to be stable and convergent to the solution. (*ii*) We derive the general condition for the global error bounds from information about the local truncation error. (*iii*) Then, the *Lax–Richtmyer stability condition for linear methods* is presented. (*iv*) Finally, we conclude with some considerations about the non-linear schemes.

Consider for simplicity a quantity q advected with constant velocity $\bar{u}$, then the dynamics is ruled by the transport equation

$$q_t + (\bar{u}q)_x = 0.$$

The real solution of this equation at time $t + \Delta t$ is obtained by translating the solution at time t of a length $\bar{u}\Delta t$. When the numerical solution evolves in one time-step, there are two possible scenarios: — the solution translates less than, or equal to, one grid cell; — the solution moves more than one grid cell. In the first case, the time step is small enough to ensure that the flux at the interface depends only on the solution at the beginning of the time step in the cells that share such interface. Figure 1.12 shows the case of $\bar{u} > 0$ and $\bar{u}\Delta t < \Delta x$. The flux at $x_{i-\frac{1}{2}}$ depends on the solution in C_{i-1}, so the solution propagates, in a single time step, less than one grid cell. In the second case, the time step is too large

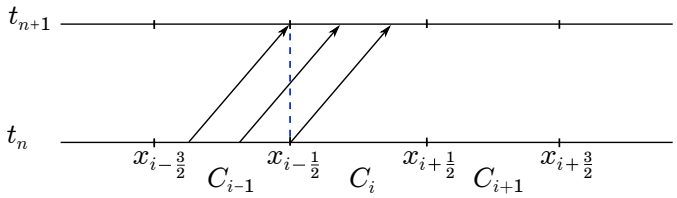


Figure 1.12: Example of flux through the interface $x_{i-\frac{1}{2}}$ that depends only on the solution Q^n_{i-1} in C_{i-1}.

and the flux at the interface depends on the value of the solution in more cells. Figure 1.13 shows the example with $\bar{u} > 0$ and $\bar{u}\Delta t > \Delta x$, in which the flux at $x_{i-\frac{1}{2}}$ depends

both on Q^n_{i-1} and on Q^n_{i-2}. We defined the numerical flux as a function $\mathcal{F}$ depending only on Q^n_{i-1} and Q^n_i, Eq. (1.64), therefore in this situation, when computing $F^n_{i-\frac{1}{2}}$, the information about the solution in cell C_{i-2} is lost, leading to instabilities. In conclusion, a

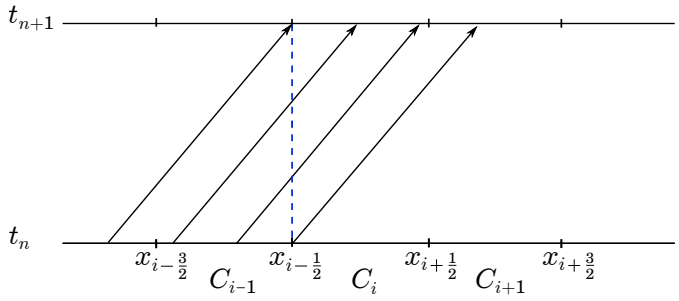


Figure 1.13: Example of flux through the interface $x_{i-\frac{1}{2}}$ that depends both on Q^n_{i-1} and on Q^n_{i-2}, the numerical solution on C_{i-1} and on C_{i-2}.

necessary (but not sufficient) condition for the stability and convergence of any numerical method for the solution of hyperbolic (more generally, time dependent) partial differential equations is the *Courant-Friedrichs-Levy condition* (see [57], [58], [168§4.4]).

> **Courant-Friederichs-Levy (CFL) condition.** A numerical method can be convergent only if its numerical domain of dependence contains the true domain of dependence of the PDE, at least in the limit as Δt and Δx go to zero.

For the simple advection equation $q_t + (\bar{u}q)_x = 0$ considered so far, the CFL condition is the following:

$$c = \left| \frac{\bar{u}\Delta t}{\Delta x} \right| \leqslant 1 \tag{1.68}$$

where c is called *Courant number*. Figure 1.12 represents a situation with $c \leqslant 1$ and the *CFL condition* is respected; conversely, in Figure 1.13 the *CFL condition* is violated because $\nu > 1$. For the generic scalar equation $\partial_t q + \partial_x f(q) = S(q)$ (where $S(q)$ consists of the source terms), the propagation speed is determined by $f'(q)$, therefore it is not homogeneous in the domain and it depends on the solution itself. Then the CFL condition relies on the following more general definition of the local Courant number:

$$c = \frac{\Delta t}{\Delta x} |f'(q)|.$$

In the latter case, the Courant number may assume different values in each cell, therefore the time step must be chosen in order that the CFL condition $c \leqslant 1$ is verified everywhere.

Now we discuss the boundedness of the global error by using the information about the local truncation error. Such analysis is easy to apply to linear methods (like the centered and upwind schemes presented respectively in §1.2.3 and §1.2.4). It leads to useful results, whereas it is hard to apply to the non-linear methods (like those that use the limiters, such as the KNP and KT schemes described in §1.2.8). The essential requirements and importance of stability may be easily seen in the following attempt to bind the global error by using a recurrence relation. At the time step n, suppose to have an approximation $\mathbf{Q}^n$ with error $\mathbf{E}^n$, so that

$$\mathbf{Q}^n = \mathbf{q}^n + \mathbf{E}^n.$$

The numerical scheme is applied to get $\mathbf{Q}^{n+1}$

$$\mathbf{Q}^{n+1} = \mathcal{N}(\mathbf{Q}^n) = \mathcal{N}(\mathbf{q}^n + \mathbf{E}^n),$$

then the global error at the new time-step is

$$\begin{aligned} \mathbf{E}^{n+1} &= \mathbf{Q}^{n+1} - \mathbf{q}^{n+1} \\ &= \mathcal{N}(\mathbf{q}^n + \mathbf{E}^n) - \mathbf{q}^{n+1} \\ &= \mathcal{N}(\mathbf{q}^n + \mathbf{E}^n) - \mathcal{N}(\mathbf{q}^n) + \mathcal{N}(\mathbf{q}^n) - \mathbf{q}^{n+1} \\ &= \left[\mathcal{N}(\mathbf{q}^n + \mathbf{E}^n) - \mathcal{N}(\mathbf{q}^n)\right] + \Delta t\, \boldsymbol{\tau}^n. \end{aligned} \tag{1.69}$$

Because of the introduction of $\mathcal{N}(\mathbf{q}^n)$, the global error at the new time step becomes the sum of two terms:

- $\mathcal{N}(\mathbf{q}^n + \mathbf{E}^n) - \mathcal{N}(\mathbf{q}^n)$, which measures the effect of the numerical method on the previous global error $\mathbf{E}^n$,
- $\Delta t\, \boldsymbol{\tau}^n$, the new one-step error introduced in this time step.

The study of the local truncation error allows us to bound the new one-step error. Stability theory is required to bound the other term, $\mathcal{N}(\mathbf{q}^n + \mathbf{E}^n) - \mathcal{N}(\mathbf{q}^n)$.

Consider the numerical schemes for which the associated numerical operator $\mathcal{N}(\cdot)$ satisfies the following property

$$\|\mathcal{N}(\mathbf{P}) - \mathcal{N}(\mathbf{Q})\| \leqslant (1 + \alpha\Delta t)\|\mathbf{P} - \mathbf{Q}\|, \tag{1.70}$$

where $\mathbf{P}$ and $\mathbf{Q}$ are any two grid functions and α is some positive constant independent of Δt as $\Delta t \to 0$; as a consequence, for those schemes the next inequality holds

$$\|\mathcal{N}(\mathbf{q}^n + \mathbf{E}^n) - \mathcal{N}(\mathbf{q}^n)\| \leqslant (1 + \alpha\Delta t)\|\mathbf{E}^n\|. \tag{1.71}$$

From this result, we obtain a boundedness on the global error

$$\begin{aligned} \left\|\mathbf{E}^N\right\| &\overset{(1.69)}{=} \left\|\left[\mathcal{N}\left(\mathbf{q}^{N-1} + \mathbf{E}^{N-1}\right) - \mathcal{N}\left(\mathbf{q}^{N-1}\right)\right] + \Delta t\, \boldsymbol{\tau}^{N-1}\right\| \\ &\overset{(1.71)}{\leqslant} (1 + \alpha\Delta t)\left\|\mathbf{E}^{N-1}\right\| + \Delta t\left\|\boldsymbol{\tau}^{N-1}\right\| \\ &\leqslant (1 + \alpha\Delta t)^N\left\|\mathbf{E}^0\right\| + \Delta t \sum_{i=0}^{N-1}(1 + \alpha\Delta t)^i\|\boldsymbol{\tau}\| \\ &= (1 + \alpha\Delta t)^N\left\|\mathbf{E}^0\right\| + \Delta t\frac{(1 + \alpha\Delta t)^N - 1}{(1 + \alpha\Delta t) - 1}\|\boldsymbol{\tau}\| \\ &\leqslant e^{\alpha T}\left(\left\|\mathbf{E}^0\right\| + \frac{1}{\alpha}\|\boldsymbol{\tau}\|\right), \end{aligned}$$

where $N\Delta t = T$ and $\|\boldsymbol{\tau}\| = \max_{0\leqslant n-1\leqslant N}\|\boldsymbol{\tau}^n\|$.

Linear methods. In the case the numerical operator $\mathcal{N}(\cdot)$ is linear, then

$$\mathcal{N}(\mathbf{q}^n + \mathbf{E}^n) = \mathcal{N}(\mathbf{q}^n) + \mathcal{N}(\mathbf{E}^n), \tag{1.72}$$

the term $\mathcal{N}(\mathbf{q}^n + \mathbf{E}^n) - \mathcal{N}(\mathbf{q}^n)$ in Eq. (1.69) simplifies as $\mathcal{N}(\mathbf{E}^n)$ and the global error at the new time step is

$$\mathbf{E}^{n+1} = \mathcal{N}(\mathbf{E}^n) + \Delta t \boldsymbol{\tau}^n.$$

Under these assumptions it is often possible to prove the inequality

$$\|\mathcal{N}(\mathbf{E})\| \leqslant (1 + \alpha\Delta t)\|\mathbf{E}\|, \tag{1.73}$$

for any arbitrary grid function $\mathbf{E}$, which implies the property (1.70); hence the global error is bounded as shown before. For the linear methods, this form of stability is generally referred to as *Lax-Richtmyer stability*.

Non-linear methods. The non-linear schemes, such as those of §1.2.8 using the geometric limiters, cannot rely upon the previous result about the global error, i.e., on the Lax-Richtmyer stability. In fact, for such schemes, the Eq. (1.73) could still hold, but it does not imply the property (1.70) that is much more difficult to prove. For the non-linear schemes, even though the CFL condition remains a necessary stability condition for the explicit schemes, other conditions might be added in the absence of a formal stability proof other conditions might be added. For the higher-order methods, it is appropriate to adopt the TV-stability associated with the TVD property that we introduce in §1.2.6.

1.2.3 Centered schemes

We begin to present examples of FVM by introducing two methods that belong to the family of the central schemes. They compute the numerical flux $F^n_{i+\frac{1}{2}}$, defined in Eq. (1.64), in a symmetric fashion with respect to the cell interface where they are evaluated. The first scheme described presents a stability lack, whereas the second one is too dissipative.

An unstable flux

A first attempt for $\mathcal{F}$ might be the simple arithmetic average of the fluxes

$$F^n_{i-\frac{1}{2}} = \mathcal{F}(Q^n_{i-1}, Q^n_i) = \frac{f(Q^n_{i-1}) + f(Q^n_i)}{2} \tag{1.74}$$

and this corresponds to the following scheme:

$$\begin{aligned} Q^{n+1}_i &= Q^n_i - \frac{\Delta t}{\Delta x}\left[\frac{f(Q^n_i) + f(Q^n_{i+1})}{2} - \frac{f(Q^n_{i-1}) + f(Q^n_i)}{2}\right] \\ &= Q^n_i - \frac{\Delta t}{2\Delta x}\Big[f(Q^n_{i+1}) - f(Q^n_{i-1})\Big]. \end{aligned} \tag{1.75}$$

This method results to be *unstable* for hyperbolic problems and cannot be used, even if the time step is small enough that the *CFL condition* is satisfied (note that the *CFL condition* is necessary for the stability, but it is not sufficient). In fact, consider the

example of the following Cauchy problem for the advection equation defined on an interval, with periodic boundary conditions:

$$\begin{aligned} &\frac{\partial q}{\partial t} + \bar{u}\frac{\partial q}{\partial x} = 0, \quad x \in [-1;1], \quad \bar{u} = 1 \\ &q(x,0) = \begin{cases} 1, & -\frac{1}{2} \leqslant x \leqslant \frac{1}{2} \\ 0, & \text{elsewhere} \end{cases} \end{aligned} \tag{1.76}$$

The exact solution is obtained by shifting the initial condition to the right. In Figure 1.14, from left to right, from up to bottom, we show the initial condition and the numerical solution after few time steps; the divergence of the solution is evident although the CFL condition (1.68) is verified, see also [168].

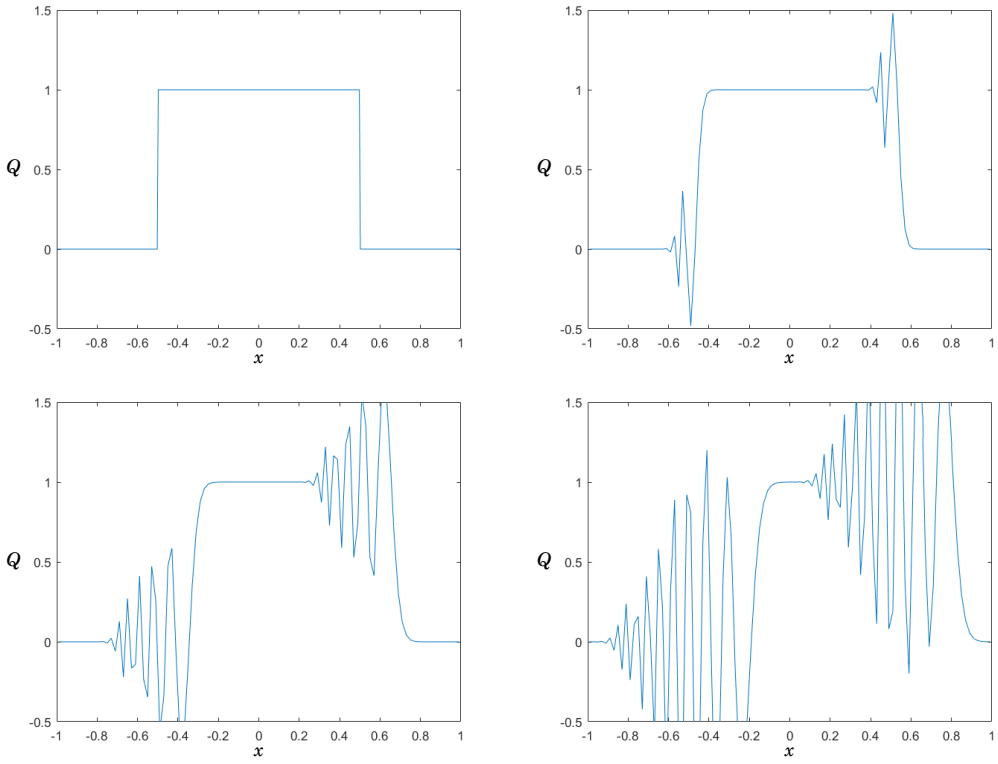


Figure 1.14: *Example of non-stability for centered schemes.* Numerical solution of the Cauchy problem of Eq. (1.76) computed by using the centered scheme, Eq. (1.74).

Lax-Friedrichs flux

The Lax-Friedrichs (LxF) scheme is obtained by replacing Q_i^n of Eq. (1.75) with the average of the numerical solution in the two neighbouring cells, namely:

$$Q_i^{n+1} = \frac{1}{2}\Big(Q_{i-1}^n + Q_{i+1}^n\Big) - \frac{\Delta t}{2\Delta x}\Big[f(Q_{i+1}^n) - f(Q_{i-1}^n)\Big]. \tag{1.77}$$

This method respects the scheme form of Eq. (1.65), i.e. the numerical flux is defined as

$$F_{i-\frac{1}{2}}^n = \mathcal{F}(Q_{i-1}^n, Q_i^n) = \frac{f(Q_{i-1}^n) + f(Q_i^n)}{2} - \frac{\Delta x}{2\Delta t}\Big(Q_i^n - Q_{i-1}^n\Big). \tag{1.78}$$

When the *CFL condition* is satisfied, the LxF method results *stable*. The numerical flux of this method is like the unstable flux of the previous scheme, Eq. (1.75), but with the additional term $\frac{\Delta x}{2\Delta t}\left(Q_i^n - Q_{i-1}^n\right)$. This term corresponds to a numerical diffusion. In fact, the scheme in Eq. (1.77) is the same we obtain applying the centered finite difference scheme to the following modification of the original problem:

$$\frac{\partial q}{\partial t} + \frac{\partial}{\partial x} f(q) = \beta \frac{\partial^2 q}{\partial x^2}, \quad \beta = \frac{(\Delta x)^2}{2\Delta t}$$

where the second order derivative term is the diffusive term. In general, stability problems occur when the numerical solution introduces a local maximum or local minimum that the method amplifies. The presence of a second-order derivative regulates the behavior of the computed solution: it assumes a negative value in the presence of a maximum, and the opposite occurs with a minimum. Therefore, the presence of the second derivative raises the points where the function is convex and lowers them where it is concave. By keeping $\Delta x/\Delta t$ fixed (whereby, fixing the Courant number), when refining the grid, the coefficient β of the diffusive term goes to zero, and the modified problem converges to the original problem.

The drawback of the diffusive behavior of the LxF scheme is that even the discontinuities proper of the solution are smoothed. The dissipating nature of LxF is shown in Figure 1.15 where we find the numerical solution of the Cauchy problem of Eq. (1.76) approximated by the LxF method, Eq. (1.77).

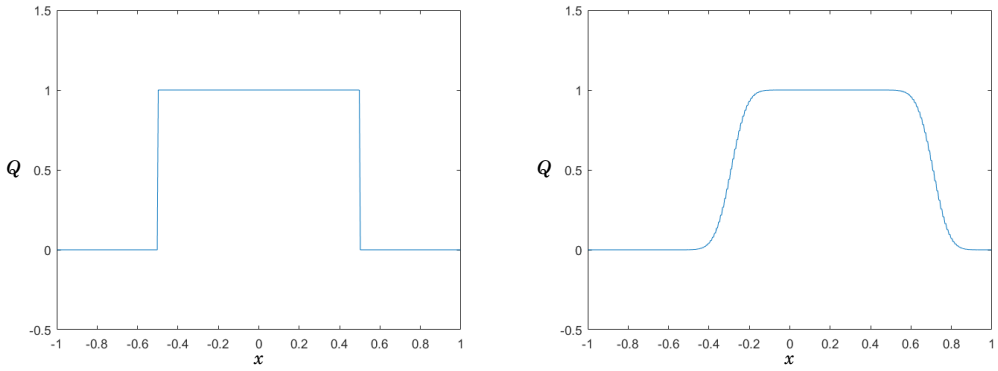


Figure 1.15: *Example of stability and numerical dissipation of the Lax-Friederichs scheme.* Numerical solution of the Cauchy problem of Eq. (1.76) computed by using Lax-Friederichs scheme, Eq. (1.77).

In conclusion, even though the LxF method is both consistent and stable, it introduces much more diffusion than is actually required and gives numerical results where the discontinuities are typically badly smeared unless a very fine grid is used.

1.2.4 Upwind schemes

The centered schemes define symmetric fluxes with respect to the interface where they are evaluated. This family of schemes does not take advantage of the intrinsic property of hyperbolic problems: the solution propagates along the characteristic curves and the propagation direction is well defined. The family of the *upwind schemes* exploits the knowledge of the hyperbolic nature of the equation to give a better definition of the numerical fluxes.

In order to introduce the upwind schemes, consider again the simple advection case with constant velocity

$$\frac{\partial q}{\partial t} + \frac{\partial(\bar{u}q)}{\partial x} = 0,$$

and assume the *CFL condition* verified, Eq. (1.68). The numerical flux $F^n_{i-\frac{1}{2}}$ depends on the velocity direction. Supposing that $\bar{u} > 0$, the characteristic curves are represented as in Figure 1.12 and the flux at the interface $x_{i-\frac{1}{2}}$ is completely determined by the value of the numerical solution in the cell C_{i-1}, i.e., Q^n_{i-1}. This suggests to express the numerical flux as

$$F^n_{i-\frac{1}{2}} = \mathcal{F}(Q^n_{i-1}, Q^n_i) = f(Q^n_{i-1}) = \bar{u}Q^n_{i-1}, \quad \bar{u} > 0.$$

From the numerical flux definition, descends the following scheme:

$$Q^{n+1}_i = Q^n_i - \frac{\bar{u}\Delta t}{\Delta x}\big(Q^n_i - Q^n_{i-1}\big), \quad \bar{u} > 0.$$

Likewise, if velocity has the opposite direction, $\bar{u} < 0$, the numerical flux is completely determined by the numerical solution at cell C_i

$$F^n_{i-\frac{1}{2}} = \mathcal{F}(Q^n_{i-1}, Q^n_i) = f(Q^n_i) = \bar{u}Q^n_i, \quad \bar{u} < 0,$$

and the scheme that descends is

$$Q^{n+1}_i = Q^n_i - \frac{\bar{u}\Delta t}{\Delta x}\big(Q^n_{i+1} - Q^n_i\big), \quad \bar{u} < 0.$$

The scheme may be rewritten as a single compact expression if we introduce the variables a^+ and a^- describing the propagation velocity to the right and to the left at the interface

$$a^+ := \max\{\bar{u}, 0\}, \quad a^- := \min\{\bar{u}, 0\}, \tag{1.79}$$

$$\implies F^n_{i+\frac{1}{2}} = a^+Q^n_i + a^-Q^n_{i+1}, \quad F^n_{i-\frac{1}{2}} = a^+Q^n_{i-1} + a^-Q^n_i; \tag{1.80}$$

hence, the numerical scheme reads as follows:

$$Q^{n+1}_i = Q^n_i - \frac{\Delta t}{\Delta x}\left[a^+\left(Q^n_i - Q^n_{i-1}\right) + a^-\left(Q^n_{i+1} - Q^n_i\right)\right].$$

In the case of an equation with a generic flux $f(q)$, the propagation velocity is provided by $f'(q)$, so that we define for every interface the right side and left side propagation velocity (that depends on the suitable cell), and consequently the numerical fluxes, as follows:

$$a^+_{i+\frac{1}{2}} := \max\{f'(Q^n_i), 0\}, \quad a^-_{i+\frac{1}{2}} := \min\{f'(Q^n_{i+1}), 0\},$$

$$\implies F^n_{i+\frac{1}{2}} = a^+_{i+\frac{1}{2}}Q^n_i + a^-_{i+\frac{1}{2}}Q^n_{i+1}, \quad F^n_{i-\frac{1}{2}} = a^+_{i-\frac{1}{2}}Q^n_{i-1} + a^-_{i-\frac{1}{2}}Q^n_i.$$

The upwind schemes are stable but diffusive. The scheme presented above is first-order accurate, but there exist more accurate upwind schemes. The second-order accurate scheme, presented in the next section, is also known as *linear upwind* and is less diffusive if compared to the first-order accurate upwind scheme.

1.2.5 Lax-Wendroff, a second order method

The second-order method named Lax-Wendroff (LxW) scheme (see [168] and pag. 23 in [226]) is born as a numerical method based on finite differences, but we show that it can also be seen as a finite volume method.

We present the simplest expression of the scheme by considering its application to the usual advective equation with the linear flux and constant velocity $\bar{u}$ (the definition of the numerical scheme for a generic flux expression $f(q)$ is presented in [168, Chapter 4.7]):

$$q_t + \bar{u} q_x = 0. \tag{1.81}$$

Given a time t_n for which we suppose to know the exact solution $q(x, t_n)$, we express the solution at time t_{n+1} by using a Taylor expansion of q with respect to the variable t starting from t_n:

$$q(x, t_{n+1}) = q(x, t_n) + \Delta t q_t(x, t_n) + \frac{1}{2}\big(\Delta t\big)^2 q_{tt}(x, t_n) + \dots \tag{1.82}$$

where $\Delta t = t_{n+1} - t_n$. The derivative with respect to t can be expressed in terms of the derivative with respect to x by using the fact that q is the solution of the PDE (1.81):

$$q_t = -\bar{u} q_x, \qquad q_{tt} = (-\bar{u} q_x)_t = \bar{u}^2 q_{xx}.$$

By substituting q_t and q_{tt} in the Taylor expansion (1.82), we obtain

$$q(x, t_{n+1}) = q(x, t_n) - \Delta t\, \bar{u} q_x(x, t_n) + \frac{1}{2}\big(\Delta t\big)^2 \bar{u}^2 q_{xx}(x, t_n) + \dots$$

When the expansion is truncated at the second order and the spatial derivatives are approximated by the following finite difference centered schemes

$$q_x(x_i, t_n) \approx \frac{Q^n_{i+1} - Q^n_{i-1}}{2\Delta x}, \qquad q_{xx}(x_i, t_n) \approx \frac{Q^n_{i+1} - 2Q^n_i + Q^n_{i-1}}{2\Delta x^2},$$

where Q^n_i is the approximation of the point-wise value $q(x_i, t_n)$ and $\Delta x = x_{i+1} - x_i$ because the LxW scheme descends from finite difference approximations, then we get the second order LxW scheme:

$$Q^{n+1}_i = Q^n_i - \frac{\Delta t}{2\Delta x}\bar{u}\big(Q^n_{i+1} - Q^n_{i-1}\big) + \left(\frac{\Delta t}{2\Delta x}\right)^2 \bar{u}^2 (Q^n_{i+1} - 2Q^n_i + Q^n_{i-1}). \tag{1.83}$$

However, the finite difference LxW scheme of Eq. (1.83) may be reinterpreted as a finite volume scheme because it is possible to find the definition of the numerical flux by reordering the terms, i.e:

$$F^n_{i-\frac{1}{2}} = \frac{1}{2}\bar{u}\big(Q^n_i + Q^n_{i-1}\big) - \frac{\Delta t}{2\Delta x}\bar{u}^2\big(Q^n_i - Q^n_{i-1}\big), \tag{1.84}$$

where now Q^n_i is the approximation of the averaged integral of $q(x, t_n)$ in the cell C_i and $\Delta x = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}$. The numerical flux expression interprets LxW as a FVM scheme.

We notice that the definition of the LxW flux, Eq. (1.84), looks like the unstable averaged flux of Eq. (1.74) plus a diffusive flux (the second-order term), but this diffusive term is chosen to exactly match what appears in the Taylor series expansion Eq. (1.82). Indeed, this shows why the averaged flux Eq. (1.74) alone is unstable: the Taylor series

expansion of the exact solution contains a diffusive term q_{xx} that is missing from the numerical method when the unstable flux is used.

The performances of LxW and upwind schemes are compared in Figure 1.16, that shows the numerical solutions of the advection equation $q_t + \bar{u}q_x = 0\,(\bar{u} = 1)$, computed with the two schemes with periodical boundary conditions. The initial condition (represented by a solid line) consists of a smooth pulse and a square-wave pulse. Because of the periodic boundary conditions and the constant velocity, the computed solution should overlap the initial conditions for every period. The results obtained by the upwind method (upper plots) show excessive dissipation. The solution computed by the LxW scheme (lower plots) captures the smooth pulse much better than in the upwind case, but, unfortunately, the square wave gives rise to an oscillatory solution. The presence of oscillations can be explained by looking at the Taylor series expansion, Eq. (1.82), as follows. The LxW scheme matches the first three terms in the series expansion, and then the dominant error is given by the next term: $q_{ttt} = -\bar{u}^3 q_{xxx}$. This term is a *dispersive* term, which leads to oscillations. In the next section, we see a different explanation for these oscillations, along with a remedy based on *limiters*.

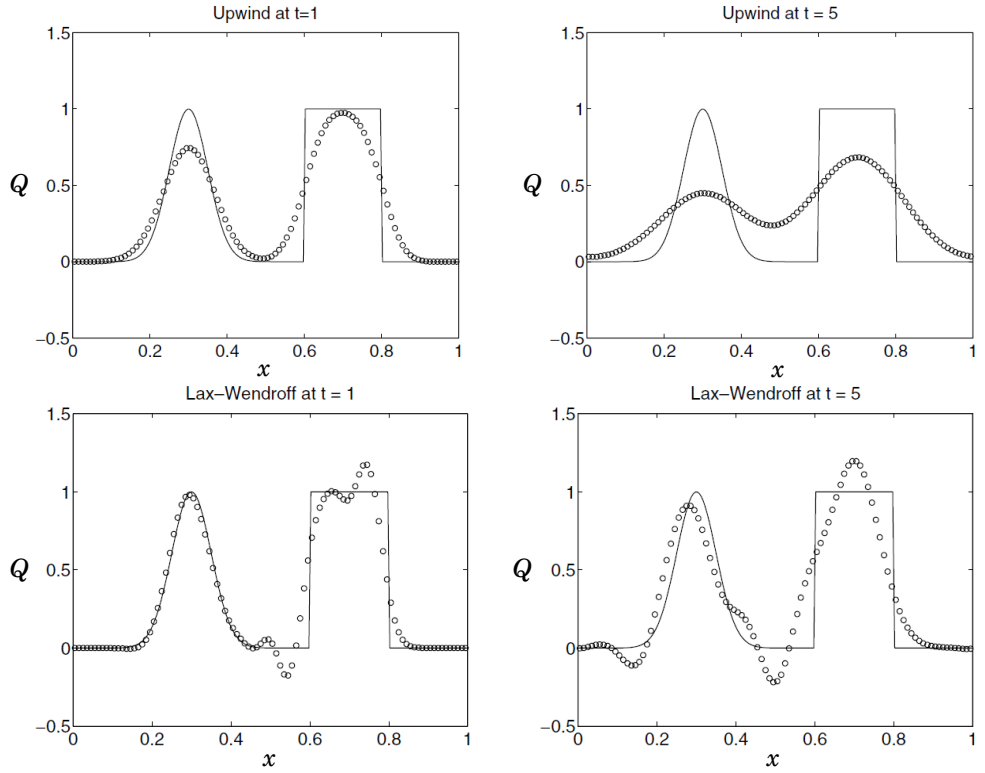


Figure 1.16: *Tests on the advection equation with upwind and Lax-Wendroff schemes (figures taken from [168, Fig. 6.1]).* The solid line represents the initial condition. Results are shown for times $t = 1$ and $t = 5$, which correspond to 1 and 5 horizontal revolutions through the domain because of the periodic boundary conditions. The results were computed by using a Courant number $\Delta t/\Delta x = 0.8$. The choice of other values gives somewhat different results, anyway the same basic behavior remains.

1.2.6 Linear reconstruction and slope limiters

The numerical schemes introduced previously to estimate the numerical flux at the interfaces, $F^n_{i+\frac{1}{2}}$, by using the values Q^n_i that approximate the averaged value of the exact

solution q over the cells C_i. The next numerical effort is to create a numerical solution that estimates the exact solution $q(x,t)$ in every point of the domain and for every time. To get this, we build such approximation $\widetilde{Q}^n(x)$ by using the cell averaged values Q_i^n to obtain a piece-wise reconstruction on the cells. Notice that there is no need for the reconstruction to be continuous: the solution reconstruction always gives useful information to write a more efficient method. In fact, if we manage to know an approximation of the solution at the interfaces, thanks to a piece-wise reconstruction, we can evaluate more precisely the fluxes at the interfaces obtaining a higher-order method. For this reason, we investigate further in a simple case what happens to the solution on the interfaces. The methods that follow are of the *Godunov's type*, namely are characterized by three steps: (1) reconstruction of the solution over the cells, (2) evolution of the solution with the computation of the numerical flux, (3) average of the evolved results over the cells. The original Godunov method used piece-wise constant reconstruction, and it was first-order. However, then the method has been generalized to polynomial piece-wise reconstruction with a higher order of accuracy. The upwind scheme may be seen as a special case of the original Godunov's scheme applied to the advective equation. Godunov's scheme is the original starting point for methods for non-linear equations [168].

The simplest kind of reconstruction is a *piece-wise constant* function, where the value Q_i^n is assumed in the whole i-th cell, i.e.

$$\widetilde{Q}^n(x) = Q_i^n, \quad \forall x \in C_i.$$

Otherwise, a *piece-wise linear* reconstruction may be adopted:

$$\widetilde{Q}^n(x) := Q_i^n + \sigma_i^n(x - x_i), \quad \forall x \in C_i, \tag{1.85}$$

where σ_i^n is the local slope. Figure 1.17 shows the two piece-wise reconstructions.

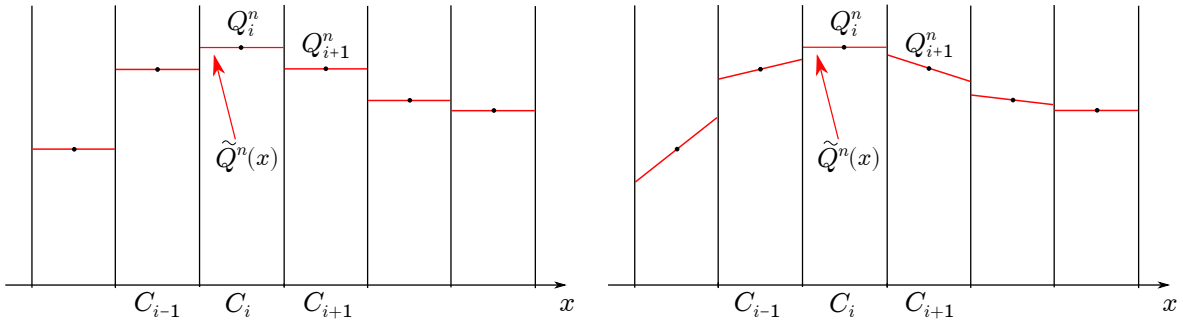


Figure 1.17: *Left:* piece-wise constant reconstruction $\widetilde{Q}^n(x)$. *Right:* piece-wise linear reconstruction $\widetilde{Q}^n(x)$.

Consider again the advection equation $q_t + \bar{u}q_x = 0$ with constant velocity $\bar{u} > 0$ and a piece-wise linear reconstruction. Recalling the definition of the numerical flux $F_{i+\frac{1}{2}}$ in Eq. (1.62), we want to estimate $f\big(q\big(x_{i+\frac{1}{2}},t\big)\big)$ that, in the actual case of advection equation, is $\bar{u}q\big(x_{i+\frac{1}{2}},t\big)$. Therefore, we want to know how the solution at the interfaces evolves in one time-step. At the interface passing by $x_{i+\frac{1}{2}}$, we consider the generic time t between t_n and t_{n+1} and find the point $(x_{i+\frac{1}{2}},t)$, Figure 1.18. As we are considering a simple advection equation with constant velocity, the solution in $(x_{i+1/2},t)$ is the same along the characteristic curve (in the plane (x,τ)) passing by it:

$$x - x_{i+\frac{1}{2}} = \bar{u}(\tau - t).$$

At time $\tau = t_n$, the characteristic curve passes by $x = x_{i+\frac{1}{2}} - \bar{u}(t - t_n)$, therefore the solutions in $(x_{i+\frac{1}{2}} - \bar{u}(t - t_n), t_n)$ and in $(x_{i+\frac{1}{2}}, t)$ are equal:

$$q\big(x_{i+\frac{1}{2}}, t\big) = q\big(x_{i+\frac{1}{2}} - \bar{u}(t - t_n)\big) \approx \widetilde{Q}^n\big(x_i + \delta x\big), \tag{1.86}$$

where we have introduced $\delta x := x_{i+\frac{1}{2}} - \bar{u}(t - t_n) - x_i$ to simplify the notation.

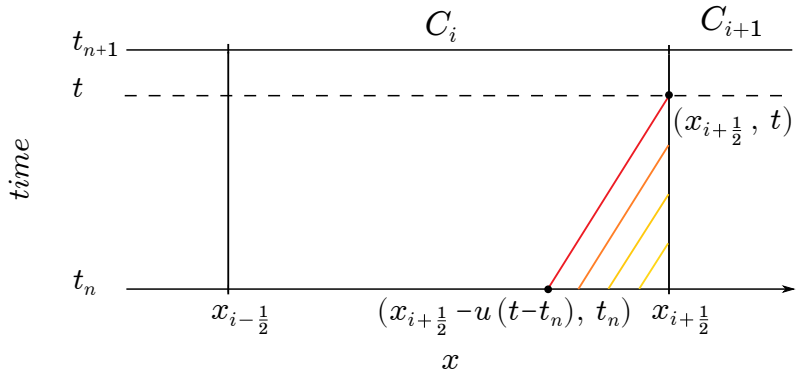


Figure 1.18: Characteristic lines passing through the interface (axes plotted: space x and time τ).

By using the linear reconstruction in the i-th cell Eq. (1.85), as shown in Figure 1.19, we obtain the value of $\widetilde{Q}^n(x_i + \delta x)$:

$$\widetilde{Q}^n(x_i + \delta x) \overset{(1.85)}{=} Q_i^n + \sigma_i^n(x_i + \delta x - x_i) = Q_i^n + \sigma_i^n \delta x. \tag{1.87}$$

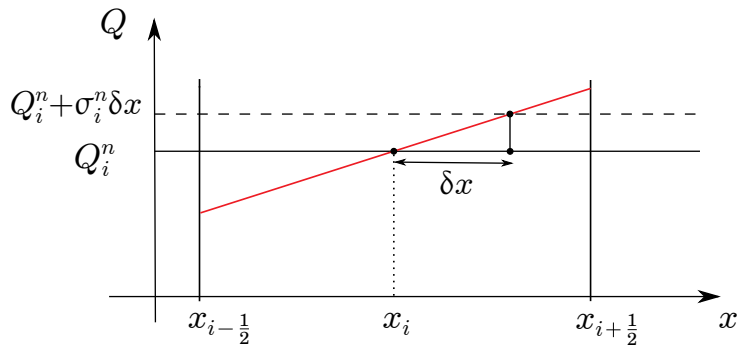


Figure 1.19: Linear reconstruction (axes plotted: x and q).

Finally, in this case of the advection equation, it is possible to evaluate the exact flux for

the approximate solution:

$$\begin{aligned} F^n_{i+\frac{1}{2}} &\overset{(1.62)}{\approx} \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} f\big(q\big(x_{i+\frac{1}{2}},t\big)\big)\mathrm{dt} \\ &= \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} \bar{u}q\big(x_{i+\frac{1}{2}},t\big)\mathrm{dt} \\ &\overset{(1.86)}{\approx} \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} \bar{u}\widetilde{Q}^n\big(x_i+\delta x\big)\mathrm{dt} \\ &\overset{(1.87)}{=} \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} \bar{u}\left(Q^n_i+\sigma^n_i\delta x\right)\mathrm{dt} \\ &= \bar{u}Q^n_i + \frac{1}{2}\bar{u}\big(\Delta x-\bar{u}\Delta t\big)\sigma^n_i. \end{aligned} \tag{1.88}$$

By comparing this flux to the LxW flux Eq. (1.84), we see that the LxW flux corresponds to the flux obtained with a linear reconstruction, where the slope is given by

$$\sigma^n_i = \frac{Q^n_{i+1}-Q^n_i}{\Delta x}. \tag{1.89}$$

Note that the approximation used in the slope reconstruction is downwind.

As seen before, the linear reconstruction of the solution is useful in the computation of the flux at the interfaces of the cells. Different definitions for the local slope σ^n_i will bring to stable or unstable methods. In fact, as said in the previous section, the LxW method works well when the initial condition is a smooth function; oscillations develop in the case of square waves instead, namely when there are discontinuities in the initial condition.

Oscillations

The piece-wise linear reconstruction might produce undesired *oscillatory* behavior where discontinuities in the initial condition are present. We try to explain why some situations give rise to such a problem and, in the following, how to avoid it by using little precautions during the linear reconstruction.

The problem of *oscillations* is related to the use of downwind approximation for the linear reconstruction. For example, suppose to have an advection equation with constant velocity $\bar{u} > 0$ and a discontinuity in the solution at time t_n defined as:

$$Q^n_j = \begin{cases} 1 & \text{if} \quad j \leqslant i, \\ 0 & \text{if} \quad j > i, \end{cases}$$

(see Figure 1.20 on the left side); if we choose the slopes downwind in each grid cell, i.e.

$$\sigma^n_i = \frac{Q^n_{i+1}-Q^n_i}{\Delta x}, \qquad i = 1,\dots,N, \tag{1.90}$$

then the reconstructed solution is constant in every cell except in the i-th, because $\sigma^n_i < 0$, hence the reconstruction in C_i is a descending line, as shown in Figure 1.20 on the right side. Unfortunately, with this reconstruction, we lost the original discontinuity of the solution, and, in addition, the reconstructed function $\widetilde{Q}^n(x)$ has an overshoot with a *new* maximum value of 1.5 regardless of Δx.

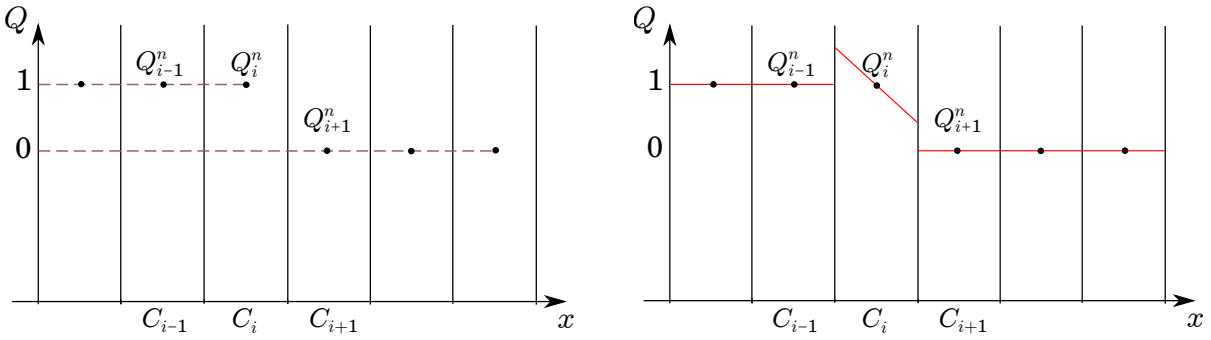


Figure 1.20: On the left, averaged cell solution at time t_n. On the right, linear reconstruction, by using downwind slope, with the introduction of a new maximum.

Since we considered a positive velocity $\bar{u} > 0$, when this numerical solution is advected of a distance $\bar{u}\Delta t$, it shifts to the right, as shown in the left picture of Figure 1.21. When we compute the average of the solution on each cell to find $\{Q_i^{n+1}\}$ for all $i = 1, \ldots, N$, in the i-th cell we will get a value greater than 1 for any Δt with $0 < \bar{u}\Delta t < \Delta x$. $\bar{u}\Delta t = \Delta x/2$ is the worst case and it implies that $Q_i^{n+1} = 1.125$; this case is represented in Figure 1.21 on the right.

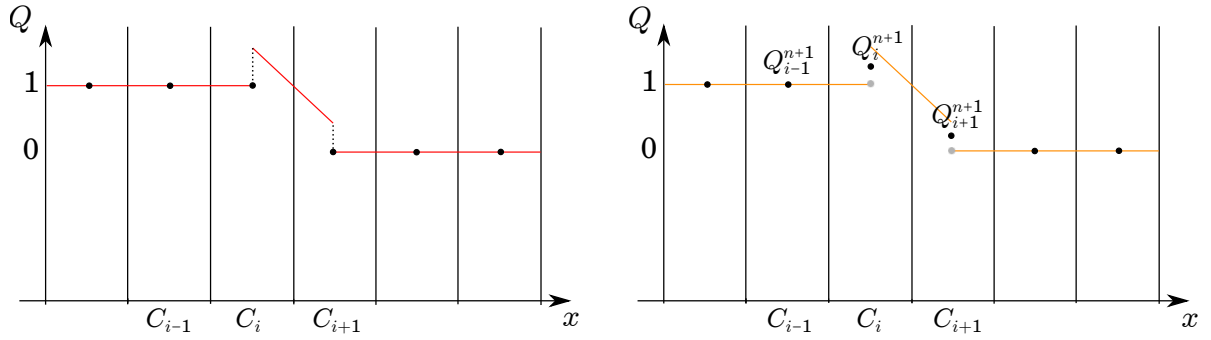


Figure 1.21: On the left, advection of the solution $\widetilde{Q}^n(x)$. On the right, averaged values of the solution on each cell at time t_{n+1}.

In the next time step this overshoot is accentuated, because in the cell C_{i-1} there is a positive slope, leading to a value of Q_{i-1}^{n+1} that is less than 1, see Figure 1.22. This oscillatory behavior increases with time.

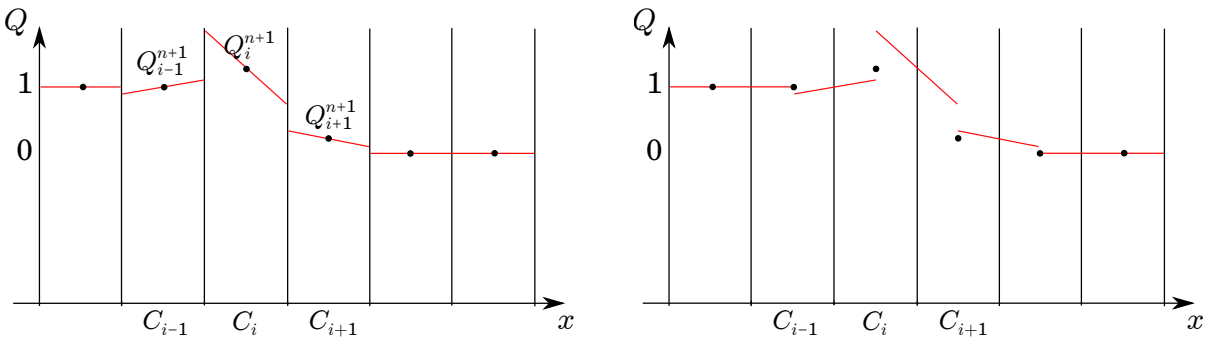


Figure 1.22: On the left, linear reconstruction of the solution $\widetilde{Q}^{n+1}(x)$. On the right, solution advected.

This problem occurs when a downwind approximation is used for the slope and $\bar{u} >$

$$\frac{\Delta t}{\Delta x} > 0.$$

The slopes proposed might be used together with the assumption of a smooth solution. There is no reason to believe that the accuracy will improve by introducing this slope near a discontinuity. On the contrary, if one of our goals is to avoid non-physical oscillations, we must set the slope to zero in the i-th cell in the example above. Any $\sigma_i^n < 0$ will lead to $Q_i^{n+1} > 1$, whereas a positive slope would not make much sense. On the other hand, we do not want to set all slopes to zero all the time, or we have the piece-wise constant reconstruction. Moreover, we will see below that even near a discontinuity, once the solution is somewhat smeared out over more than one cell, introducing non-zero slopes may help to keep the solution from smearing out too far, and hence will significantly increase the resolution and keep discontinuities fairly sharp, as long as care is taken to avoid oscillations. This suggests that we must pay attention to how the solution behaves near the i-th cell in choosing our formula for σ_i^n. Where the solution is smooth, we want to choose something like a downwind slope, as in the Lax-Wendroff slope. Near a discontinuity, we want to limit this slope by using a value that is smaller in magnitude to avoid oscillations. Methods based on this idea are known as *slope-limiter methods*.

The methods that adopt the slope limiters are second or higher-order and are not linear schemes. Before introducing them, we present an important result from Godunov for the development of non-oscillatory schemes. Godunov proved this theorem as a Ph.D. student at Moscow State University, [104], [105].

Theorem 1.5 (Godunov's order barrier theorem)**.** *Linear numerical schemes for solving PDEs, having the property of not generating new extrema (non-oscillatory scheme), can be at most first-order accurate.*

A consequence of this result is that high-order non-oscillatory schemes cannot be linear. Notice that the Lax-Wendroff scheme respects this theorem, being it linear, second-order, but generates oscillations. Also, the upwind scheme follows the theorem being a linear scheme, non-oscillatory, and first order.

Total variation diminishing

How much should we limit the slope? Ideally, we would like to have a mathematical prescription that will allow us to use downwind slope (1.90) whenever possible while guaranteeing that no non-physical oscillations will arise. To achieve this, we need a way to *measure oscillations in the solution*. This is provided by the notion of the ***total variation*** of a function: for a grid function $Q = \{Q_i\}_i$ we define

$$TV(Q) := \sum_i |Q_i - Q_{i-1}|. \tag{1.91}$$

The *total variation* at the time t_{n+1} of the previous example of Figures 1.20 and 1.21 is larger than the *total variation* at the time t_n: $TV(Q^{n+1}) > TV(Q^n)$, where $Q^n = \{Q_i^n\}_i$. For an advection problem, the exact solution propagates with velocity $\bar{u}$ and its shape does not change with time. Consequently, we request the *total variation* to be the same at each time step. As seen before, the LxW scheme applied to an advection problem with a discontinuous initial condition does not satisfy this request. For more general equations, we want that a numerical scheme does not increase the *total variation*.

Definition 1.9. *A method is said* TVD (total variation diminishing) *if, for any initial condition Q^n, the solution Q^{n+1} evaluated by such method satisfies:*

$$TV\left(Q^{n+1}\right) \leqslant TV\left(Q^{n}\right).$$

Geometric limiters

One choice of slope that gives second-order accuracy for smooth solutions while still satisfying the *TVD* property is the ***minmod slope*** defined by

$$\sigma_i^n = \text{minmod}\left(\frac{Q_i^n - Q_{i-1}^n}{\Delta x}, \frac{Q_{i+1}^n - Q_i^n}{\Delta x}\right) \tag{1.92}$$

where *minmod* is a function of two arguments

$$\text{minmod}\,(a,b) = \begin{cases} a & \text{if} \quad |a| < |b| \quad \text{and} \quad ab > 0 \\ b & \text{if} \quad |a| > |b| \quad \text{and} \quad ab > 0 \\ 0 & \text{if} \quad ab \leqslant 0. \end{cases} \tag{1.93}$$

Using the *minmod slope* for the example in Figure 1.23 on the left (where dashed lines are the arguments used by the *minmod* function), we obtain the reconstruction represented on the right. Moreover, we can see that no spurious maximum has been added in the reconstruction.

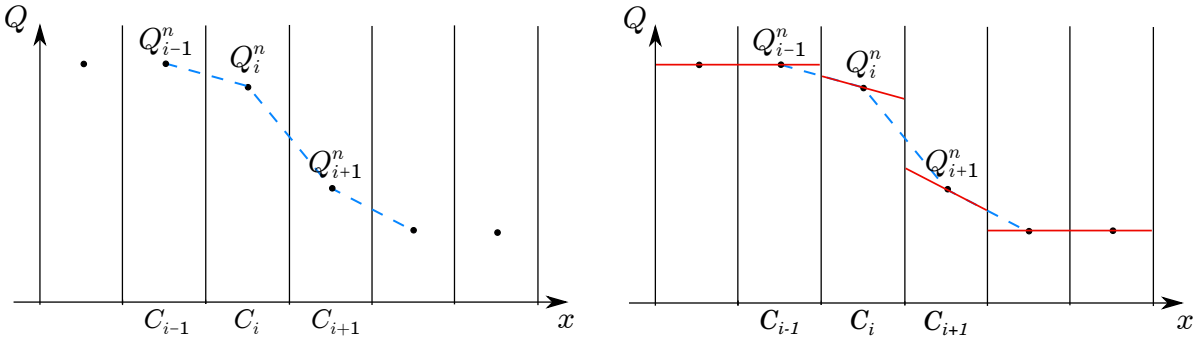


Figure 1.23: Linear reconstruction using the *minmod slope*.

Another choice of limiter that still gives second order accuracy for smooth solutions, is the so-called ***superbee limiter*** introduced by Roe:

$$\sigma_i^n = \text{maxmod}\left(\sigma_i^{(1)}, \sigma_i^{(2)}\right)$$

where

$$\sigma_i^{(1)} = \text{minmod}\left(\frac{Q_i^n - Q_{i-1}^n}{\Delta x}, 2\frac{Q_{i+1}^n - Q_i^n}{\Delta x}\right),$$

$$\sigma_i^{(2)} = \text{minmod}\left(2\frac{Q_i^n - Q_{i-1}^n}{\Delta x}, \frac{Q_{i+1}^n - Q_i^n}{\Delta x}\right),$$

$$\text{maxmod}(a,b) = \begin{cases} b & \text{if} \quad |a| < |b| \quad \text{and} \quad ab > 0 \\ a & \text{if} \quad |a| > |b| \quad \text{and} \quad ab > 0 \\ 0 & \text{if} \quad ab \leqslant 0; \end{cases}$$

each one-sided slope is compared with twice the opposite one-sided slope, then the maxmod function selects the argument with larger modulus, for details see [168]. In Figure 1.24 we have the comparison between minmod and superbee use.

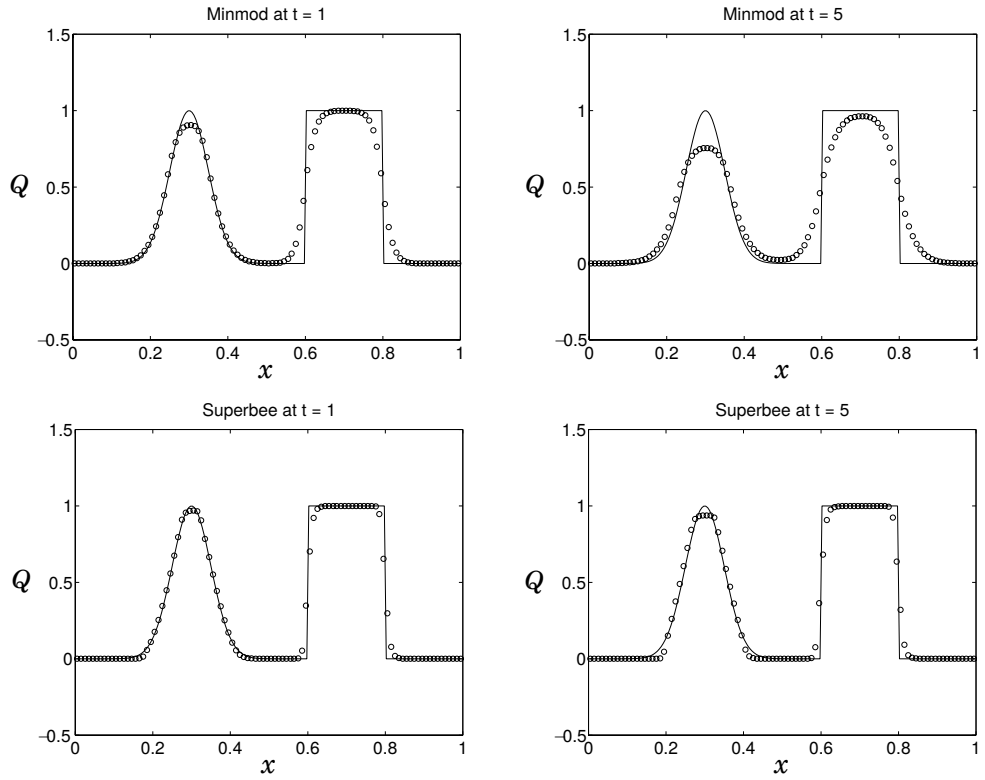


Figure 1.24: *Test on the advection equation with high-resolution methods that adopt the minmod and superbee limiters respectively (figure taken from [168, Ch. 6.3, page 104]).* The solid line represents the initial condition of a smooth and a square pulses. Results are shown for times $t = 1$ and $t = 5$, corresponding to 1 and 5 horizontal revolutions through the domain because of the periodic boundary conditions.

1.2.7 Flux limiters

We consider again the LxW scheme and rewrite the numerical flux for an advective equation with a constant velocity $\bar{u}$ in the following alternative form:

$$F^n_{i-\frac{1}{2}} = a^- Q^n_i + a^+ Q^n_{i-1} + \frac{1}{2}|\bar{u}| \left(1 - \frac{|\bar{u}|\Delta t}{\Delta x}\right) \left(Q^n_i - Q^n_{i-1}\right)$$

where $a^\pm$ follow the definition of Eq. (1.79). Notice that the first two terms on the right-hand side correspond to the upwind scheme (Eq. (1.80)), which is first order, and the last term is a second-order correction. We observed that the upwind scheme treats the discontinuity better, whereas the LxW scheme works better with the smooth parts of the solution. Therefore we want to introduce a coefficient for the second-order term (the last term) acting as a *flux limiter*, able to switch from the first-order to the second-order schemes accordingly to the smoothness of the solution and without introducing oscillations in the linear reconstruction.

The limiters on the flux may be obtained starting from the geometric limiter on the slopes of the linear reconstructions. The flux limiters "measure" the smoothness of the solution. Generally, the flux limiters are applied to the jump in the solution (ΔQ) instead of the approximated slope ($\Delta Q/\Delta x$) as done so far. We also denote as $\delta^n_{i-\frac{1}{2}}$ the limited version of the jump at the interface, $\Delta Q^n_{i-\frac{1}{2}} = Q^n_i - Q^n_{i-1}$. Denoting the flux limiter as

$\phi(\cdot)$, then $\delta^n_{i-\frac{1}{2}}$ is defined as follows:

$$\delta^n_{i-\frac{1}{2}} = \phi\left(\theta^n_{i-\frac{1}{2}}\right)\Delta Q^n_{i-\frac{1}{2}},$$

where the parameter $\theta^n_{i-\frac{1}{2}}$ measures the smoothness of the solution near the interface:

$$\theta^n_{i-\frac{1}{2}} = \begin{cases} \dfrac{\Delta Q^n_{i-\frac{3}{2}}}{\Delta Q^n_{i-\frac{1}{2}}}, & \text{if } \bar{u} > 0, \\ \dfrac{\Delta Q^n_{i+\frac{1}{2}}}{\Delta Q^n_{i-\frac{1}{2}}}, & \text{if } \bar{u} < 0. \end{cases}$$

The flux limiter definition and the smoothness study depend on the sign of velocity. In fact, if we have $\bar{u} > 0$, then the solution propagates to the right; therefore, it is important to check the smoothness of the solution on the left of the interface. We expect that $\theta^n_{i-\frac{1}{2}} \approx 1$ where the solution is smooth, whereas near discontinuities the value of $\theta^n_{i-\frac{1}{2}}$ may be far from 1.

The flux-limiter function $\phi(\theta)$ assumes values dependent on the solution smoothness. By setting $\phi(\theta) = 1$ for all θ produces the LxW method, whereas $\phi(\theta) = 0$ gives the upwind scheme. More generally, the aim is to set a limiter function ϕ that assumes values close to 1 for $\theta \approx 1$, reducing the jump where data are not smooth.

It is possible to translate the various slope limiters into flux-limiter functions, obtaining the flux-limiter functions reported in Table 1.1.

Linear methods	
upwind:	$\phi(\theta) = 0$
LxW:	$\phi(\theta) = 1$
High-resolution limiters	
minmod:	$\phi(\theta) = \max(0, \min(1, \theta))$
superbee:	$\phi(\theta) = \max(0, \min(1, 2\theta), \min(2, \theta))$

Table 1.1: Flux-limiter functions and classification of the methods. For more schemes see [168].

For simple limiters such as minmod, it is easy to check that the TVD condition of Definition 1.9 is satisfied. Instead, for more complicated limiters, this condition can be more difficult to be proven.

1.2.8 Kurganov-Noelle-Petrova scheme

In this section, we present a second-order TVD scheme belonging to the family of the Godunov's type schemes (Reconstruct-Evolve-Average schemes) introduced in §1.2.6. This scheme has the advantages of simplicity, high accuracy, and a semi-discrete expression. Thanks to the use of linear reconstruction with slope limiter, there is less numerical dissipation at the non-smooth parts of the solution; therefore, the accuracy is higher. This method was described by Kurganov et al. [160] as an improvement of the scheme introduced by Kurganov and Tadmor [159]. In §1.2.8 the differences between them are

shown. Moreover, Kurganov and Petrova [158] adapted the KNP scheme to the Saint-Venant system of shallow-water equations. In §2.1 we derive a depth-averaged model and in §3 we present the numerical scheme which is based on such adapted method.

The family of the high-resolution Godunov-type schemes may also be described and characterized by projection-evolution steps. Starting with the cell averages at time level t_n, one reconstructs a piecewise interpolating polynomial of degree $r-1$, where r is the formal order of the scheme (in our discussion $r=2$), which is evolved to the next time level t_{n+1}, and then it is projected onto a space of piecewise constants.

The KNP scheme follows a *central-upwind* approach. In fact, we can say that the scheme is of *central* Godunov-type because the evolution step employs the integration over the region of influence of the interface points. It also has an *upwind* nature since one-sided information is used to estimate the region width.

As anticipated previously, the KNP scheme is based on a *semi-discrete* formulation, namely the temporal derivative is retained, resulting in a formulation like

$$\frac{d}{dt}Q_i(t) = -\frac{F_{i+\frac{1}{2}}(t) - F_{i-\frac{1}{2}}(t)}{\Delta x}. \tag{1.94}$$

The great advantage of this formulation is allowing the adoption of the ODE solvers of higher order, such as Runge-Kutta, so that a second-order spatial scheme may be associated with a second-order temporal scheme. The semi-discrete formulation is obtained passing to the limit $\Delta t \to 0$ whereas Δx is left fixed. In §1.2.8, we show that this method does not possess a semi-discrete formulation, although the LxF scheme has a relationship with it.

At each time step of the scheme, we do these four passages:

1. Piecewise linear *reconstruction* of the solution;
2. *Estimation of the speed* of the solution through the interfaces;
3. Computation of the *numerical flux*;
4. *Evolution* of the solution at the following time step.

Reconstruction. Suppose to be at time level $t = t_n$ and to know the cell average values $\{Q_i^n\}_i$, then the discontinuous piece-wise linear reconstruction is calculated:

$$\widetilde{Q}^n(x) = Q_i^n + \sigma_i^n(x - x_i), \quad x_{i-\frac{1}{2}} < x < x_{i+\frac{1}{2}}, \quad \forall i, \tag{1.95}$$

where, according with previous arguments, the slopes $\{\sigma_i^n\}_i$ should be computed by using a geometric limiter such as those seen in §1.2.6.

The reconstructed solution may be discontinuous at the interfaces, as shown in Figure 1.25, then we define $Q^L_{i+\frac{1}{2}}$ and $Q^R_{i+\frac{1}{2}}$ as the left and right sided reconstructed solution point value respectively, namely:

$$\begin{aligned} Q^R_{i+\frac{1}{2}} &:= \lim_{x \to x^+_{i+\frac{1}{2}}} \widetilde{Q}^n(x) = Q^n_{i+1} - \frac{\Delta x}{2}\sigma^n_{i+1}, \\ Q^L_{i+\frac{1}{2}} &:= \lim_{x \to x^-_{i+\frac{1}{2}}} \widetilde{Q}^n(x) = Q^n_i + \frac{\Delta x}{2}\sigma^n_i. \end{aligned} \tag{1.96}$$

Note that the quantities $Q^R_{i+\frac{1}{2}}$ and $Q^L_{i+\frac{1}{2}}$ depend on the time-step t_n, but, for simplicity, we suppress the time-dependence in our notation.

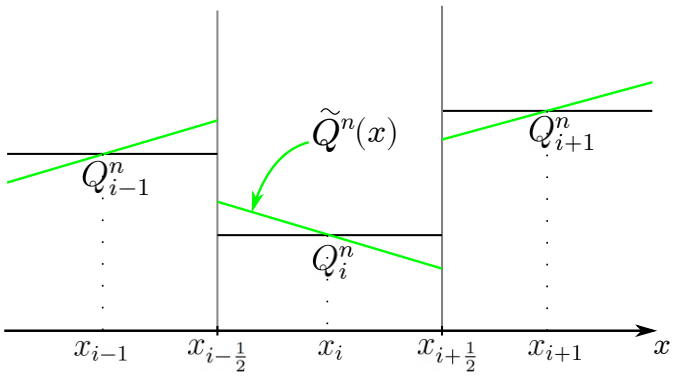


Figure 1.25: *Black* lines represent the cell-averaged values of the solution, while the *green* lines depicts a linear reconstruction of the solution over the cells.

Estimation of the local speed. The solution propagates at the interfaces with right-sided and left-sided local speeds which must be estimated. In the case of the simple advective equation with constant velocity $q_t + \bar{u}q_x = 0$, the propagation velocity is exactly $\bar{u}$ and it is the same at each interface. In the case of a generic non-linear equation $\partial_t q + \partial_x f(q) = 0$, the spatial derivative is expanded according with the chain rule and the equation is rewritten in the quasi-linear form

$$\frac{\partial q}{\partial t} + \frac{df(q)}{dq}\frac{\partial q}{\partial x} = 0,$$

and the coefficient df/dq results to be the local velocity of propagation, which depends on the solution and, therefore, is potentially different at each interface.

The discontinuity at the interface moves to the right at most with a velocity that corresponds to the maximum of the positive propagation velocities; on the opposite, the solution moves on the left at most with a velocity that is the minimum of the negative propagation velocities. Hence we are interested in these two values:

$$\begin{aligned} a^+_{i+\frac{1}{2}} &:= \max\left\{\frac{df}{dq}\left(Q^R_{i+\frac{1}{2}}\right), \frac{df}{dq}\left(Q^L_{i+\frac{1}{2}}\right), 0\right\}, \\ a^-_{i+\frac{1}{2}} &:= \min\left\{\frac{df}{dq}\left(Q^R_{i+\frac{1}{2}}\right), \frac{df}{dq}\left(Q^L_{i+\frac{1}{2}}\right), 0\right\}. \end{aligned} \tag{1.97}$$

For each interface, we define the two points that contain the region of influence (that is, the region where q could propagate according to the values (1.97)) at time t_{n+1}:

$$\begin{aligned} x^n_{i+\frac{1}{2},r} &:= x_{i+\frac{1}{2}} + \Delta t a^+_{i+\frac{1}{2}}, \\ x^n_{i+\frac{1}{2},l} &:= x_{i+\frac{1}{2}} + \Delta t a^-_{i+\frac{1}{2}}. \end{aligned} \tag{1.98}$$

By reminding that $a^-_{i+\frac{1}{2}} \leqslant 0$, we know that these two points are on the right and left side of $x_{i+\frac{1}{2}}$ respectively, as in Figure 1.26.

Numerical flux. By using the notation $x^n_{i+\frac{1}{2},r/l}$ of Eq. (1.98), we separate the regions in which the solution undergoes the effects of discontinuities propagation at the interfaces from the parts that are not affected by that. Therefore, we consider the following non-equal intervals

$$J_i := \left[x^n_{i-\frac{1}{2},r}, x^n_{i+\frac{1}{2},l}\right], \quad \text{and} \quad J_{i+\frac{1}{2}} := \left[x^n_{i+\frac{1}{2},l}, x^n_{i+\frac{1}{2},r}\right], \tag{1.99}$$

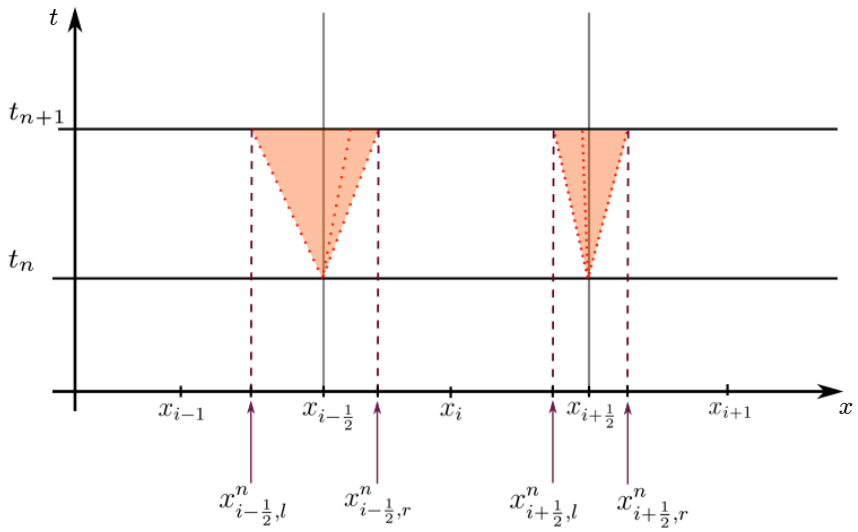


Figure 1.26: On the plane (x,t), the *red points* represent the possible characteristic lines (*Riemann fan*) at the interfaces at the time t_n, whereas the *red triangles* denote the regions of influence.

and the solution is smooth in the first interval, whereas it is non-smooth in the second one. Notice that these intervals depend on the time-step t_n, but we have neglected this dependency in the notation.

In order to compute the cell averages of the exact solution at time t_{n+1}, we distinguish between the solution averaged over the smooth regions, named ω_i^{n+1}, and the solution averaged over the non-smooth region, called $\omega_{i+\frac{1}{2}}^{n+1}$, by integrating over the corresponding domains in (1.99), see Figure 1.27:

$$\omega_i^{n+1} \approx \frac{1}{|J_i|}\int_{J_i} q(x,t_{n+1})dx,$$

$$\omega_i^{n+1} := \frac{1}{|J_i|}\left\{\int_{J_i}\widetilde{Q}^n(x)dx - \int_{t_n}^{t_{n+1}}\left[f\left(q\left(x^n_{i+\frac{1}{2},l},t\right)\right) - f\left(q\left(x^n_{i-\frac{1}{2},r},t\right)\right)\right]dt\right\}, \quad (1.100)$$

and

$$\omega_{i+\frac{1}{2}}^{n+1} \approx \frac{1}{\left|J_{i+\frac{1}{2}}\right|}\int_{J_{i+\frac{1}{2}}} q(x,t_{n+1})dx,$$

$$\omega_{i+\frac{1}{2}}^{n+1} := \frac{1}{\left|J_{i+\frac{1}{2}}\right|}\left\{\int_{J_{i+\frac{1}{2}}}\widetilde{Q}^n(x)dx - \int_{t_n}^{t_{n+1}}\left[f\left(q\left(x^n_{i+\frac{1}{2},r},t\right)\right) - f\left(q\left(x^n_{i+\frac{1}{2},l},t\right)\right)\right]dt\right\}. \quad (1.101)$$

Thanks to the definition of the piece-wise linear reconstruction $\widetilde{Q}^n(x)$, Eq. (1.95), the spatial integrals can be calculated explicitly. The discretization of the flux integrals may require an appropriate quadrature formula, since the solution is smooth along the line segments $\left(x^n_{i+\frac{1}{2},l},t\right)$ and $\left(x^n_{i+\frac{1}{2},r},t\right)$, with $t_n \leqslant t < t_{n+1}$. At this stage, we realize the solution at time level t_{n+1} in terms of the approximate cell averages ω_i^{n+1} and $\omega_{i+\frac{1}{2}}^{n+1}$. These averages spread over a non-uniform grid which is over-sampled by twice the number of

the original cells. The last thing to do is to *convert* these averages back into the original grid.

To do this, from the averaged values ω_i^{n+1} and $\omega_{i+\frac{1}{2}}^{n+1}$ we build a piece-wise linear reconstruction on the non-uniform grid at time t_{n+1} by computing new values for the slopes with geometric limiters (introduced in §1.2.6): we denote as $\widetilde{\omega}_i^{n+1}(x)$ the linear reconstruction for the interval J_i, and as $\widetilde{\omega}_{i+\frac{1}{2}}^{n+1}(x)$ the reconstruction for $J_{i+\frac{1}{2}}$. Hence the global piece-wise linear function on the staggered cells takes the following expression:

$$\widetilde{\omega}^{n+1}(x) := \sum_i \left(\widetilde{\omega}_i^{n+1}(x)\chi_{J_i}(x) + \widetilde{\omega}_{i+\frac{1}{2}}^{n+1}(x)\chi_{J_{i+\frac{1}{2}}}(x) \right), \tag{1.102}$$

where $\chi_I(x)$ is the characteristic function of the interval I. Thus, with the computation of the cell averages at the next time level, namely by projecting $\widetilde{\omega}^{n+1}$ back on the original grid, the scheme is completely constructed:

$$Q_i^{n+1} = \frac{1}{\Delta x} \int_{C_i} \widetilde{\omega}^{n+1}(x)dx \,, \forall i. \tag{1.103}$$

Noticing that the (possibly non-trivial) linear reconstruction $\widetilde{\omega}_i^{n+1}(x)$ is averaged out on the interval J_i for the computation of Q_i^{n+1} in (1.103), any linear choice leads again to the value ω_i^{n+1} because the region is smooth:

$$\frac{1}{|J_i|} \int_{J_i} \widetilde{\omega}_i^{n+1}(x)dx = \omega_i^{n+1}. \tag{1.104}$$

From this observation, we conclude that we may consider a constant reconstruction over J_i. Figure 1.27 represents these choices for the reconstructions over the different types of intervals.

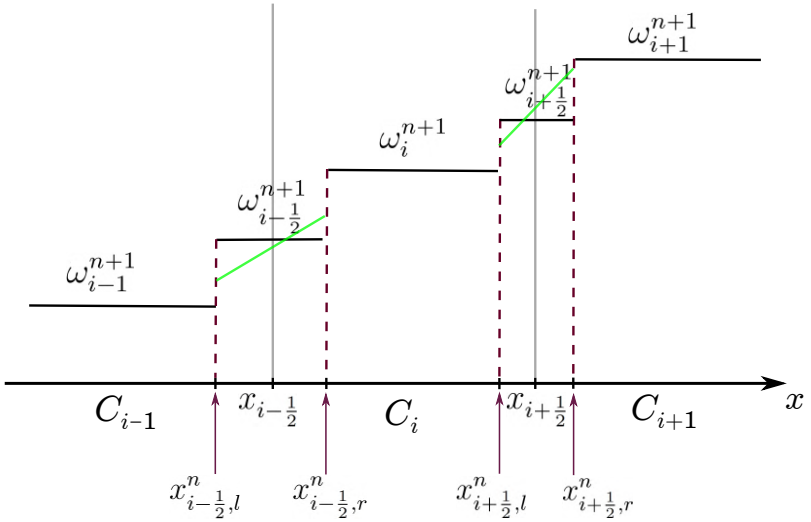


Figure 1.27: The horizontal *black* lines represent the piecewise constant reconstruction obtained with ω_i^{n+1} and $\omega_{i+\frac{1}{2}}^{n+1}$. The *green* lines depict the linear reconstruction $\widetilde{\omega}_{i+\frac{1}{2}}^{n+1}$.

The previous treatment brings a fully discrete Godunov-type central-upwind scheme that can be derived explicitly, similarly to the derivation of other central schemes [156, 159]. However, we are not interested in this formulation, and we present the derivation of a semi-discrete expression of the scheme in the form (1.94). The semi-discrete formulation

is obtained by expressing the time derivative $\frac{d}{dt}Q_i(t)$ as the limit, for Δt that goes to zero, of the discretized term as follows:

$$\begin{aligned}\frac{d}{dt}Q_i(t) &= \lim_{\Delta t\to 0}\frac{Q_i^{n+1}-Q_i^n}{\Delta t}\\ &\overset{(1.103)}{=} \lim_{\Delta t\to 0}\frac{1}{\Delta t}\left[\frac{1}{\Delta x}\int_{C_i}\widetilde{\omega}^{n+1}(x)dx - Q_i^n\right].\end{aligned} \tag{1.105}$$

The integral over the whole cell C_i splits over three intervals: $C_i\cap J_{i-\frac{1}{2}}$, J_i, and $C_i\cap J_{i+\frac{1}{2}}$. We suppose that the slopes of the linear reconstructions $\widetilde{\omega}_{i+\frac{1}{2}}^{n+1}$ are uniformly bounded, independently of Δt. Therefore, as the width of $J_{i+\frac{1}{2}}$ is equal to $\left(a_{i+\frac{1}{2}}^{+}-a_{i+\frac{1}{2}}^{-}\right)\Delta t$, we obtain

$$\widetilde{\omega}_{i+\frac{1}{2}}^{n+1}(x) = \omega_{i+\frac{1}{2}}^{n+1}+\mathcal{O}(\Delta t),\quad \forall x\in J_{i+\frac{1}{2}}. \tag{1.106}$$

By using the previous equation, we evaluate separately the integrals over the three intervals, and we get

$$\int_{C_i\cap J_{i-\frac{1}{2}}}\widetilde{\omega}^{n+1}(x)dx \overset{(1.102)}{=} \int_{x_{i-\frac{1}{2}}}^{x_{i-\frac{1}{2},r}}\widetilde{\omega}_{i-\frac{1}{2}}^{n+1}(x)dx \overset{(1.98),(1.106)}{=} a_{i-\frac{1}{2}}^{+}\Delta t\,\omega_{i-\frac{1}{2}}^{n+1}+\mathcal{O}((\Delta t)^2)\,, \tag{1.107}$$

$$\int_{C_i\cap J_{i+\frac{1}{2}}}\widetilde{\omega}^{n+1}(x)dx \overset{(1.102)}{=} \int_{x_{i+\frac{1}{2},l}}^{x_{i+\frac{1}{2}}}\widetilde{\omega}_{i+\frac{1}{2}}^{n+1}(x)dx \overset{(1.98),(1.106)}{=} a_{i+\frac{1}{2}}^{-}\Delta t\,\omega_{i+\frac{1}{2}}^{n+1}+\mathcal{O}((\Delta t)^2)\,, \tag{1.108}$$

$$\int_{J_i}\widetilde{\omega}^{n+1}(x)dx \overset{(1.102)}{=} \int_{x_{i-\frac{1}{2},r}}^{x_{i+\frac{1}{2},l}}\widetilde{\omega}_{i}^{n+1}(x)dx \overset{(1.104)}{=} |J_i|\omega_i^{n+1}. \tag{1.109}$$

By using the results of Eqs. (1.107–1.109), the time derivative of the Eq. (1.105) may be expressed as follows

$$\frac{d}{dt}Q_i(t) = \frac{a_{i-\frac{1}{2}}^{+}}{\Delta x}\lim_{\Delta t\to 0}\omega_{i-\frac{1}{2}}^{n+1}+\lim_{\Delta t\to 0}\frac{1}{\Delta t}\left(\frac{|J_i|}{\Delta x}\omega_i^{n+1}-Q_i^n\right)+\frac{a_{i+\frac{1}{2}}^{-}}{\Delta x}\lim_{\Delta t\to 0}\omega_{i+\frac{1}{2}}^{n+1}\,. \tag{1.110}$$

We compute the three limits separately.

Using the definition of $\omega_{i+\frac{1}{2}}^{n+1}$ in Eq. (1.101), we obtain that the third limit may be expressed as follows

$$\lim_{\Delta t\to 0}\omega_{i+\frac{1}{2}}^{n+1} = \frac{a_{i+\frac{1}{2}}^{+}Q_{i+\frac{1}{2}}^{R}-a_{i+\frac{1}{2}}^{-}Q_{i+\frac{1}{2}}}{a_{i+\frac{1}{2}}^{+}-a_{i+\frac{1}{2}}^{-}}-\frac{f(Q_{i+\frac{1}{2}}^{R})-f(Q_{i+\frac{1}{2}}^{L})}{a_{i+\frac{1}{2}}^{+}-a_{i+\frac{1}{2}}^{-}}, \tag{1.111}$$

and a similar expression descends also for the first limit. Instead, by using the definition of ω_i^{n+1} of Eq. (1.100), the second limit results as:

$$\lim_{\Delta t\to 0}\frac{1}{\Delta t}\left(\frac{x_{i+\frac{1}{2},l}^{n}-x_{i-\frac{1}{2},r}^{n}}{\Delta x}\omega_i^{n+1}-Q_i^n\right) = \frac{a_{i+\frac{1}{2}}^{-}Q_{i+\frac{1}{2}}^{L}-a_{i-\frac{1}{2}}^{+}Q_{i-\frac{1}{2}}^{R}}{\Delta x}-\frac{f(Q_{i+\frac{1}{2}}^{L})-f(Q_{i-\frac{1}{2}}^{R})}{\Delta x}. \tag{1.112}$$

Finally, a substitution of Eqs. (1.112) and (1.111) in Eq. (1.110) results in the KT semi-discrete central-upwind scheme, which can be written in the canonical conservative form by considering a generic time t instead of t_n:

$$\frac{d}{dt}Q_i(t) = -\frac{F_{i+\frac{1}{2}}(t)-F_{i-\frac{1}{2}}(t)}{\Delta x}, \tag{1.113}$$

where the numerical flux $F_{i+\frac{1}{2}}$ is defined as

$$F_{i+\frac{1}{2}}(t) = \frac{a^+_{i+\frac{1}{2}} f\left(Q^L_{i+\frac{1}{2}}(t)\right) - a^-_{i+\frac{1}{2}} f\left(Q^R_{i+\frac{1}{2}}(t)\right)}{a^+_{i+\frac{1}{2}} - a^-_{i+\frac{1}{2}}} + \frac{a^+_{i+\frac{1}{2}} a^-_{i+\frac{1}{2}}}{a^+_{i+\frac{1}{2}} - a^-_{i+\frac{1}{2}}} \left[Q^R_{i+\frac{1}{2}}(t) - Q^L_{i+\frac{1}{2}}(t)\right]; \quad (1.114)$$

notice that the first term on the right-hand side is a weighted average of the evaluated fluxes with the speeds of propagation because the total flux at the interface depends both on the left-going and on the right-going fluxes, each weighted with the maximum speed in that direction.

Evolution. The semi-discrete scheme (1.113)-(1.115) is a time-dependent ODE (Ordinary Differential Equation) which can be solved by any stable ODE solver that retains the order of the spatial accuracy of the semi-discrete scheme. If the forward Euler scheme is adopted, the final discretization results in the next equation

$$Q^{n+1}_i = Q^n_i - \Delta t \left(\frac{F^n_{i+\frac{1}{2}} - F^n_{i-\frac{1}{2}}}{\Delta x} \right).$$

Instead, if the backward Euler method is applied, we obtain the following discretization:

$$Q^{n+1}_i = Q^n_i - \Delta t \left(\frac{F^{n+1}_{i+\frac{1}{2}} - F^{n+1}_{i-\frac{1}{2}}}{\Delta x} \right).$$

Kurganov-Tadmor and Lax-Friedrichs schemes as particular cases of KNP

The KNP scheme described before is a direct improvement of the method proposed by Kurganov and Tadmor [159]. The two schemes differ in the usage of the propagation velocity at the interface; in fact, Kurganov and Tadmor [159] derived their numerical flux KT by considering only the maximum velocity of propagation instead of Eq. (1.97):

$$a_{i+\frac{1}{2}} := \max\left\{ \left| \frac{df}{dq}\left(Q^R_{i+\frac{1}{2}}\right) \right|, \left| \frac{df}{dq}\left(Q^L_{i+\frac{1}{2}}\right) \right| \right\}.$$

Therefore, by imposing that the left and right propagation velocities have the same magnitude of $a_{i+\frac{1}{2}}$, namely that

$$a^+_{i+\frac{1}{2}} = a_{i+\frac{1}{2}}, \quad a^-_{i+\frac{1}{2}} = -a_{i+\frac{1}{2}},$$

and by replacing these values in the expression (1.114) of the KNP flux, then the KT flux descends:

$$F_{i+\frac{1}{2}}(t) = \frac{f\left(Q^L_{i+\frac{1}{2}}(t)\right) + f\left(Q^R_{i+\frac{1}{2}}(t)\right)}{2} - \frac{a_{i+\frac{1}{2}}}{2}\left[Q^R_{i+\frac{1}{2}}(t) - Q^L_{i+\frac{1}{2}}(t)\right]. \quad (1.115)$$

Moreover, the scheme reduces to the first-order LxF method in the case that the piece-wise linear reconstruction $\widetilde{Q}^n$ is replaced by a piece-wise constant reconstruction, and the forward Euler scheme is adopted. In fact, in the case of constant reconstruction, the values at the interfaces are the same as the averaged values

$$Q_{i+\frac{1}{2}} \to Q^n_i, \quad Q^R_{i+\frac{1}{2}} \to Q^n_{i+1},$$

and the propagation velocities are assumed to be

$$a_{i+\frac{1}{2}} = \frac{\Delta x}{\Delta t};$$

as a consequence, the numerical flux of Eq. (1.115) reduces to the LxF numerical flux defined in Eq. (1.78).

We add the further observation about the LxF scheme that it does not admit a semi-discretization expression, differently from the KT and KNP schemes. To prove it, we pass to the limit $\Delta t \to 0$ in the expression of the scheme (as we did in Eq. (1.105) for the KNP scheme), keeping constant the value of Δx. By rearranging the terms in Eq. (1.77, the LxF scheme writes according to the next expression

$$\frac{Q_i^{n+1} - Q_i^n}{\Delta t} = \frac{f(Q_{i-1}^n) - f(Q_{i+1}^n)}{2\Delta x} + \frac{1}{2\Delta t}\Big(Q_{i+1}^n + Q_{i-1}^n - 2Q_i^n\Big). \tag{1.116}$$

Passing to the limit $\Delta t \to 0$, the left-hand side term results in the exact temporal derivative $\frac{d}{dt}Q_i(t)$, whereas the second term on the right-hand side, that is related to dissipation, increases and tends to infinity. This observation offers a second point of view on the dissipating behavior of the LxF scheme. When the CFL condition is respected with $\Delta t \ll \Delta x$, the dissipation is excessive and produces smearing of the solution.

1.2.9 Numerical discretization of a system of hyperbolic PDEs

In the previous sections, from §1.2.1 to §1.2.8, we presented the theory and schemes of the FVMs applied to the numerical solution of a single scalar equation in a 1D context. Nevertheless, we know that CFD often requires solving systems of equations, even in a multidimensional context. Hence, in the present section, we give hints to connect what was described previously and translate it into this more general context.

In §1.1.6 we wrote in vectorial notation a generic (possibly non-linear) *hyperbolic system of PDEs* (see Definition 1.4) as

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{q}) = \mathbf{S}(\mathbf{q}) \quad \Longleftrightarrow \quad \frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{f}^{(1)}(\mathbf{q})}{\partial x} + \frac{\partial \mathbf{f}^{(2)}(\mathbf{q})}{\partial y} + \frac{\partial \mathbf{f}^{(3)}(\mathbf{q})}{\partial z} = \mathbf{S}(\mathbf{q}),$$

where $\mathbf{q} = [q_1, \ldots, q_N]^T$ is the vector of the conservative variables, $\mathbf{f} = \left(\mathbf{f}^{(1)}, \mathbf{f}^{(2)}, \mathbf{f}^{(3)}\right)$ is the vector of the fluxes with the components in each direction (if we consider the case of a 3 dimensional problem), and $\mathbf{S}(\mathbf{q})$ is the vector of the source terms.

Suppose to apply a uniform discretization of the spatial domain obtaining hexahedral cells $C_{i,j,k}$ (centered at the point (x_i, y_j, z_k)), then the finite volume method described in §1.2.1 is applied similarly to the Eqs. (1.55). One defines $\mathbf{Q}_{i,j,k}^n$ as the approximation of the average solution inside the cell $C_{i,j,k}$ at the time t_n:

$$\mathbf{Q}_{i,j,k}^n \approx \frac{1}{Vol(C_{i,j,k})} \int_{C_{i,j,k}} \mathbf{q}(\mathbf{x}, t_n) dV. \tag{1.117}$$

and also the numerical fluxes at the interfaces of the three directional fluxes:

$$\mathbf{F}^{(1),n}_{i+\frac{1}{2},j,k} \approx \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}}\int_{y_{j-\frac{1}{2}}}^{y_{j+\frac{1}{2}}}\int_{z_{k-\frac{1}{2}}}^{z_{k+\frac{1}{2}}} \mathbf{f}^{(1)}\big(\mathbf{q}\big(x_{i+\frac{1}{2}},y,z,t\big)\big)dz\,dy\,dt,$$

$$\mathbf{F}^{(2),n}_{i,j+\frac{1}{2},k} \approx \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}}\int_{y_{j-\frac{1}{2}}}^{y_{j+\frac{1}{2}}} \mathbf{f}^{(2)}\big(\mathbf{q}\big(x,y_{i+\frac{1}{2}},z,t\big)\big)dx\,dz\,dt,$$

$$\mathbf{F}^{(3),n}_{i,j,k+\frac{1}{2}} \approx \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}}\int_{y_{j-\frac{1}{2}}}^{y_{j+\frac{1}{2}}} \mathbf{f}^{(3)}\big(\mathbf{q}\big(x,y,z_{i+\frac{1}{2}},t\big)\big)dx\,dy\,dt.$$

The FVM for systems has the generic expression, which corresponds to Eq. (1.63) of the scalar case, as follows:

$$\mathbf{Q}^{n+1}_{i,j,k} = \mathbf{Q}^{n}_{i,j,k} - \frac{\Delta t}{\Delta x}\left[\mathbf{F}^{(1),n}_{i+\frac{1}{2},j,k} - \mathbf{F}^{(1),n}_{i-\frac{1}{2},j,k}\right] - \frac{\Delta t}{\Delta y}\left[\mathbf{F}^{(2),n}_{i,j+\frac{1}{2},k} - \mathbf{F}^{(2),n}_{i,j-\frac{1}{2}.k}\right] - \frac{\Delta t}{\Delta z}\left[\mathbf{F}^{(3),n}_{i,j,k+\frac{1}{2}} - \mathbf{F}^{(3),n}_{i,j,k-\frac{1}{2}}\right].$$

For example, the numerical flux definition of the centered schemes seen previously in Eqs. (1.74), (1.78) may be adapted in a straightforward way.

Since we are mostly interested in the use of the KNP scheme, we consider the semi-discrete formalization:

$$\frac{d\mathbf{Q}_{i,j,k}(t)}{dt} = -\frac{\mathbf{F}^{(1)}_{i+\frac{1}{2},j,k}(t) - \mathbf{F}^{(1)}_{i-\frac{1}{2},j,k}(t)}{\Delta x} - \frac{\mathbf{F}^{(2)}_{i,j+\frac{1}{2},k}(t) - \mathbf{F}^{(2)}_{i,j-\frac{1}{2},k}(t)}{\Delta y} - \frac{\mathbf{F}^{(3)}_{i,j,k+\frac{1}{2}}(t) - \mathbf{F}^{(3)}_{i,j,k-\frac{1}{2}}(t)}{\Delta z}.$$

Next, we show how to calculate the first term on the right-hand side of the equation, referring to the flow along the x-direction. Then, the remaining two terms can be determined with the same procedure.

According to the numerical scheme KNP, a linear reconstruction at the interface is computed for each direction. We denote as $\widetilde{\mathbf{Q}}(x,y,z)$ the piece-wise linear reconstruction of the numerical solution which is the multidimensional and vector analogue of Eq. (1.95). Then the values assumed at the interfaces along the x-direction are:

$$\mathbf{Q}^{E}_{i+\frac{1}{2},j,k} := \lim_{x\to x^{+}_{i+\frac{1}{2}}} \widetilde{\mathbf{Q}}(x,y_j,z_k) = \mathbf{Q}_{i+1,j,k} - \frac{\Delta x}{2}\boldsymbol{\sigma}^{(1)}_{i+1,j,k},$$

$$\mathbf{Q}^{W}_{i+\frac{1}{2},j,k} := \lim_{x\to x^{-}_{i+\frac{1}{2}}} \widetilde{\mathbf{Q}}(x,y_j,z_k) = \mathbf{Q}_{i,j,k} + \frac{\Delta x}{2}\boldsymbol{\sigma}^{(1)}_{i,j,k},$$

where we have neglected the time dependence. These definitions correspond to Eq. (1.96). Also, notice that $\boldsymbol{\sigma}^{(1)}_{i,j,k}$ is the vector of the slopes of the variables in $\mathbf{Q}_{i,j,k}$ along the x-direction and the possible adopted limiter is applied component-wise.

The velocity of propagation of these solutions at the interfaces depend on the maximum and minimum eigenvalues of the Jacobian matrix of the flux function $\mathbf{f}^{(1)}$ evaluated at the interface, then we define

$$a^{+}_{i+\frac{1}{2},j,k} := \max\left\{\lambda_N\left(\frac{d\mathbf{f}^{(1)}}{d\mathbf{q}}\left(\mathbf{Q}^{W}_{i+\frac{1}{2},j,k}\right)\right), \lambda_N\left(\frac{d\mathbf{f}^{(1)}}{d\mathbf{q}}\left(\mathbf{Q}^{E}_{i+\frac{1}{2},j,k}\right)\right), 0\right\},$$

$$a^{-}_{i+\frac{1}{2},j,k} := \min\left\{\lambda_1\left(\frac{d\mathbf{f}^{(1)}}{d\mathbf{q}}\left(\mathbf{Q}^{W}_{i+\frac{1}{2},j,k}\right)\right), \lambda_1\left(\frac{d\mathbf{f}^{(1)}}{d\mathbf{q}}\left(\mathbf{Q}^{E}_{i+\frac{1}{2},j,k}\right)\right), 0\right\}.$$

Finally, we define the numerical flux of the KNP scheme along the x-direction in the multidimensional and vector case (the analog of Eq. (1.115) for the scalar 1-dimensional scheme):

$$\mathbf{F}^{(1)}_{i+\frac{1}{2},j,k}(t) = \frac{a^+_{i+\frac{1}{2},j,k}\mathbf{f}^{(1)}\left(\mathbf{Q}^W_{i+\frac{1}{2},j,k}(t)\right) - a^-_{i+\frac{1}{2}}\mathbf{f}^{(1)}\left(\mathbf{Q}^E_{i+\frac{1}{2},j,k}(t)\right)}{a^+_{i+\frac{1}{2},j,k} - a^-_{i+\frac{1}{2}j,k}} + \frac{a^+_{i+\frac{1}{2},j,k}a^-_{i+\frac{1}{2},j,k}}{a^+_{i+\frac{1}{2},j,k} - a^-_{i+\frac{1}{2},j,k}}\left[\mathbf{Q}^E_{i+\frac{1}{2},j,k}(t) - \mathbf{Q}^W_{i+\frac{1}{2},j,k}(t)\right].$$

Stability and CFL condition

In §1.2.2.2 we discussed about stability and introduced the CFL condition (a necessary condition for stability) for a scalar equation, we explore a little what we can say in the case of a system.

In the case of a one dimensional system of PDEs such as $\partial_t\mathbf{q} + \partial_x\mathbf{f}(\mathbf{q}) = \mathbf{S}(\mathbf{q})$, the physical speeds of propagation are determined by the eigenvalues of the flux Jacobian matrix $\dfrac{d\mathbf{f}}{d\mathbf{q}}$, named as $\lambda_1 \leqslant \cdots \leqslant \lambda_N$, then the Courant number is defined as follows in terms the eigenvalue with the maximum magnitude

$$c := \frac{\Delta t}{\Delta x}\max_p|\lambda_p|,$$

because the highest propagation speed imposes the most restrictive condition on the time step.

In the case that the problem is multidimensional, then the flux function $\mathbf{f}$ has components $\left(\mathbf{f}^{(1)}, \mathbf{f}^{(2)}, \mathbf{f}^{(3)}\right)$, one for each direction. Considers the matrices $\dfrac{d\mathbf{f}^{(i)}}{d\mathbf{q}}$, $i = 1, \ldots, 3$ which are the Jacobians of the flux functions $\mathbf{f}^{(i)}(\mathbf{q})$ with eigenvalues $\lambda^{(i)}_1 \leqslant \cdots \leqslant \lambda^{(i)}_N$. For each direction, we define the Courant number $c^{(i)}$ associated as

$$c^{(i)} = \frac{\Delta t}{\Delta x}\max_p\left|\lambda^{(i)}_p\right|, \quad i = 1, \ldots, 3$$

then the CFL condition to respect is that

$$c := \max\left(c^{(1)}, c^{(2)}, c^{(3)}\right) \leqslant 1.$$

Notice that the Courant number c, and hence the CFL condition, might assume different values in each cell because of the change in the values of the eigenvalues. For this reason, the time-step Δt must be chosen such that the CFL condition $c \leqslant 1$ is verified contemporary in each cell.

In the discussion about stability in §1.2.2.2, we stated that the high-order schemes applied to scalar equations need to respect the TVD property (described in §1.2.6) as a necessary requirement for stability. Unfortunately, the TV notion does not extend to the systems of non-linear equations. In general, the exact solution of systems itself does not satisfy a TVD property in any reasonable sense, so we expect the numerical solution to do the same. Moreover, to our knowledge, no convergence proof is available for the Godunov scheme even in the more classic and simple systems cases (such as the shallow-water

equations or the Euler equations), as stated in [168]. There is a sort of connection between the knowledge of the solution's existence and defining stability conditions. Consider, for example, that Courant, Friedrichs, and Levy determined the CFL condition while were studying the existence of the solution, see [57, 58]. Therefore, it seems obvious that there is no stability condition for the Godunov schemes in the absence of convergence results. Despite this lack of rigorous results, both the Godunov method and its high-resolution variations generally succeed in practice and are widely used.

1.3 Computational setting with OpenFOAM

A part of our work leans on the OpenFOAM software, in fact, we developed a solver in the OpenFOAM framework in order to solve numerically our 3D model, therefore we give some fundamental information about it. OpenFOAM (Open Field Operation And Manipulation) is a free, open source software for Computational Fluid Dynamics (CFD) produced primarily by OpenCFD Ltd since 2004. It is widely used across different areas of engineering and science, from both commercial and academic organisations. OpenFOAM is a `C++` library using the object-oriented programming (OOP). OpenFOAM has an extensive range of features to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to acoustics, solid mechanics and electromagnetism. Also, OpenFOAM permits High-Performance Computing (HPC) because it offers the possibility to launch and handle the computations in parallel. Finally, OpenFOAM has frequent updates and there is a profitable correspondence between the communities of users and of developers that helps to improve the performance and to fix possible bugs of the software. The website https://www.openfoam.com/ provides further information about OpenFOAM, there are also a User's Guide, a Tutorial Guide and a wiki site with the links to the repositories.

In the following, after a description of the spatial domain discretization into a mesh in §1.3.1, we present how OpenFOAM discretizes a generic transport equation (§1.3.2). OpenFOAM treats and discretizes in a specific manner each different type of term present in a PDE and then produces an algebraic equation to solve for each cell of the discretized domain. Gathering together such algebraic equations, OpenFOAM assembles an algebraic system, as shown in §1.3.3, and solves it.

We underline that the whole section is devoted to describe the application of the FVM to the numerical solution, in the context of the OpenFOAM software, of a single PDE. When having to solve a system of coupled PDEs, more attention and a specific strategy are required in order to preserve the coupling, but in this section we neglect this aspect (it will be treated in §4.1 by means of the segregated strategy).

For more details on the way the equations are defined in OpenFOAM and how we can add an equation to an already existing system, see §4.5 and §A respectively.

1.3.1 Computational domain and variable arrangement

The spatial domain, over which the PDE is defined, is discretized as a mesh of $\mathcal{N}$ elements, referred to as cells or control volumes. Each control volume V_P of the computational domain is identified by its **centroid**, say it P, and its volume is V. The position of the centroid P is $\mathbf{x}_P$ such that:

$$\mathbf{x}_P \in V_P : \int_{V_P} (\mathbf{x} - \mathbf{x}_P)\, dV = 0. \tag{1.118}$$

The vector that connects the centroid P of V_P to the centroid N of a neighboring cell V_N is named $\mathbf{d}$ (or $\mathbf{d}_{PN}$), see Figure 1.28.

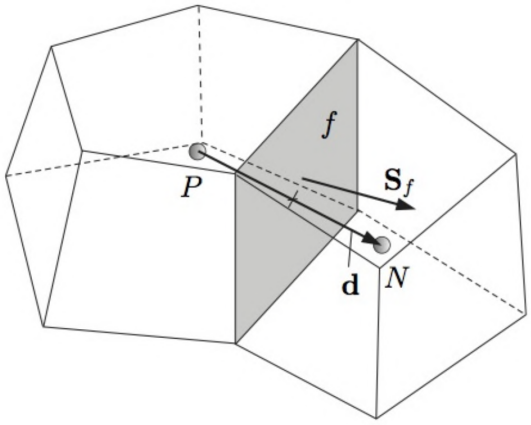


Figure 1.28: Example of two neighboring control volumes P and N. The face f shared by them is *owned* by the control volume P and points outward; $\mathbf{S}_f$ is the surface area vector.

The boundary of a control volume is constituted by a finite number of faces $f \subset \partial V_P$. An *internal* face is shared between two neighboring cells, whereas a *boundary* face belongs only to one control volume and it constitutes the boundary of the domain. Considering an internal face, even though it is shared by two cells, only one control volume is denoted as the *owner* of the face, while the other control volume is labeled as *neighbor*. The cells at the boundary are the only *owners* of their boundary faces. For each face f, the position of its central point (denoted as f, like the face itself) is $\mathbf{x}_f$:

$$\mathbf{x}_f \in f \subset \partial V_P : \int_f (\mathbf{x} - \mathbf{x}_f) \cdot d\mathbf{S} = 0. \tag{1.119}$$

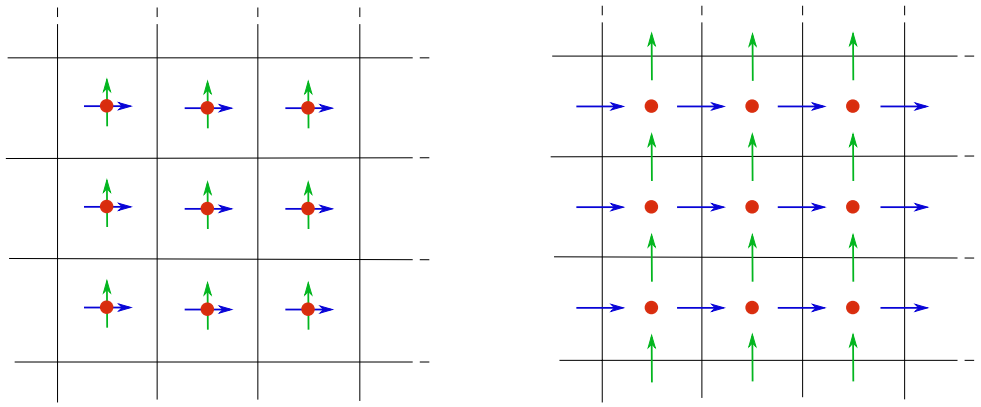
A surface area vector $\mathbf{S}_f$ is then associated to each face, it is positioned at the face center $\mathbf{x}_f$, it is normal to the face, its magnitude is equal to the area of the face and it points outward to the owner cell it belongs to (see Figure 1.28); in the case of a boundary face, the surface area vector $\mathbf{S}_f$ points outward the computational domain.

For our case study, the mesh is always "still", in the sense that it may be refined at run-time, but for sure the cells do not move with respect to the fixed coordinate system that we consider.

Cell-centered co-located arrangement. For the storage of the values of the discretized variables (e.g., pressure or velocity) OpenFOAM uses a cell-centered co-located arrangement. This means that the values of all the discretized variables are stored at the same positions (*co-located arrangement*) that are the center of the control volumes (*cell-centered*), as represented in Figure 1.29a. Because of this type of arrangement, we must compute interpolations every time we need the values at the faces. In OpenFOAM, the value at time t_n of any stored variables, say q, is its cell average

$$Q_P^n \approx \frac{1}{V_P} \int_{V_P} q(\mathbf{x}, t_n) dV.$$

The alternative to the co-located memorization is the staggered arrangement, represented symbolically in Figure 1.29b, for which the scalar variables (like pressure, density, temperature, energy) are stored at the cell center of the control volumes, whereas velocity or momentum variables are referred to the face centers.



(a) *Co-located arrangement.* (b) *Staggered arrangement.*

Figure 1.29: The *red* dots refer to pressure p, the *blue* and *green* arrows refer to the velocity components u and v respectively.

1.3.2 Transport equation discretization

The system of governing equations that OpenFOAM solves is the system of Navier-Stokes Eqs. (1.37) or its variations like Eqs. (1.49) possibly enriched by other transport equations. We describe how OpenFOAM deals with a generic transport equation, see §1.1.4, for a scalar conservative property (that might be the density, temperature, energy, enthalpy, and more), but the discussion may be appropriately extended also to vector quantities (like velocity and momentum).

Let q be a conservative scalar quantity advected with the flow of velocity $\mathbf{u}$ according with the following standard transport equation:

$$\frac{\partial q}{\partial t} + \nabla \cdot (q\mathbf{u}) - \nabla \cdot (\Gamma \nabla q) = S(q), \tag{1.120}$$

where the terms are, in order from left to right, the temporal derivative, the convection term, the diffusion term (Γ is a diffusion coefficient) and S is a source term that might depend on q. Notice that the equation is a second-order PDE because the diffusive term involves the second derivative in space. In general, for a good approximation, the discretization order in space must be equal to or higher than the order of the discretized equation. We use a piece-wise linear approximation of the exact solution, name it $\widetilde{Q}(\mathbf{x}, t)$, to guarantee the required second-order accuracy. The approximating function is such that at the cell centroids P at the time step t_n it approximates the averaged value of the solution over the considered V_P control volume, namely $\widetilde{Q}(\mathbf{x}_P, t_n) = Q_P^n$. Being $\widetilde{Q}$ a linear approximation in each control volume, we have:

$$\widetilde{Q}(\mathbf{x}, t_n) = Q_P^n + (\mathbf{x} - \mathbf{x}_P) \cdot \nabla \widetilde{Q}(\mathbf{x}_P, t_n), \ \forall \mathbf{x} \in V_P. \tag{1.121}$$

Similarly, a piece-wise linear approximation is assumed also on the faces for any vector quantity $\mathbf{a}$:

$$\widetilde{\mathbf{A}}(\mathbf{x}, t_n) = \mathbf{A}_f^n + (\mathbf{x} - \mathbf{x}_f) \cdot \nabla \widetilde{\mathbf{A}}(\mathbf{x}_f, t_n), \ \forall \mathbf{x} \in f, \tag{1.122}$$

where $\mathbf{A}_f^n = \widetilde{Q}(\mathbf{x}_f, t_n)$.

OpenFOAM, using the Finite Volume method, demands that Eq. (1.120) is verified

over each control volume V_P in the integral form of the PDE, that is:

$$\int_t^{t+\Delta t}\left[\int_{V_P}\frac{\partial q}{\partial t}dV+\int_{V_P}\nabla\cdot(q\mathbf{u})\,dV-\int_{V_P}\nabla\cdot(\Gamma\nabla q)\,dV\right]dt=\int_t^{t+\Delta t}\left(\int_{V_P}S(q)dV\right)dt. \tag{1.123}$$

The discretization of Eq. (1.123) requires the use of the Gauss Theorem and peculiar treatments for every term.

1.3.2.1 Space discretization

We consider Eq. (1.123) temporarily neglecting the integral over the time interval $[t, t+\Delta t]$. The partial time derivative moves out of the integral over the control volume because we assumed to deal with a discretized domain that does not depend on time. The difficulty to handle the volume integrals of divergence and gradient operators is overcome by using the Gauss theorem (often called divergence theorem). In fact, the Gauss theorem permits to convert the volume integral over the control volume V_P of most spatial derivative terms into integrals over the cell boundaries ∂V_P. By adopting the divergence theorem on Eq. (1.123) we obtain

$$\frac{\partial}{\partial t}\int_{V_P}qdV+\int_{\partial V_P}(q\mathbf{u})\cdot d\mathbf{S}-\int_{\partial V_P}(\Gamma\nabla q)\cdot d\mathbf{S}=\int_{V_P}S(q)dV,$$

where $d\mathbf{S}$ represents an infinitesimal surface element, which associated vector points outward normal on ∂V_P. The boundary of the control volume is constituted by a finite number of faces, so each surface integral over ∂V_P is a finite sum of the surface integrals over each face f:

$$\frac{\partial}{\partial t}\int_{V_P}qdV+\sum_{f\in\partial V_P}\left[\int_f(q\mathbf{u})\cdot d\mathbf{S}\right]-\sum_{f\in\partial V_P}\left[\int_f(\Gamma\nabla q)\cdot d\mathbf{S}\right]=\int_{V_P}S(q)dV \tag{1.124}$$

Reminding that the piece-wise linear function $\widetilde{Q}$ approximates the solution q,see Eq. (1.121), we can discretize the differential terms. For example, the volume integral of the quantity q is:

$$\begin{aligned}\int_{V_P}qdV&\approx\int_{V_P}\widetilde{Q}dV\\&\overset{(1.121)}{=}\int_{V_P}\left[Q_P+(\mathbf{x}-\mathbf{x}_P)\cdot(\nabla\widetilde{Q})_P\right]dV\\&=Q_P\int_{V_P}dV+(\nabla\widetilde{Q})_P\cdot\int_{V_P}(\mathbf{x}-\mathbf{x}_P)dV\\&\overset{(1.118)}{=}Q_PV_P\end{aligned} \tag{1.125}$$

where the last equality is due to the definition of centroid. From Eq. (1.125) descends a straightforward discretization of the **transient term** in Eq. (1.124):

$$\frac{\partial}{\partial t}\int_{V_P}qdV\overset{(1.125)}{\approx}\frac{\partial}{\partial t}\left(Q_PV_P\right). \tag{1.126}$$

In a similar way, the surface integral of the vector quantity $\mathbf{a}$ through the face f is approximated as follows:

$$\begin{aligned}\int_f \mathbf{a}\cdot d\mathbf{S} &\approx \int_f \widetilde{\mathbf{A}}\cdot d\mathbf{S}\\ &\overset{(1.122)}{=} \int_f \Big[\mathbf{A}_f + (\mathbf{x}-\mathbf{x}_f)\cdot(\nabla\widetilde{\mathbf{A}})_f\Big]\cdot d\mathbf{S}\\ &= \mathbf{A}_f\cdot\int_f d\mathbf{S} + (\nabla\widetilde{\mathbf{A}})_f\cdot\int_f(\mathbf{x}-\mathbf{x}_f)\cdot d\mathbf{S}\\ &\overset{(1.119)}{=} \mathbf{A}_f\cdot\mathbf{S}_{f,P}\end{aligned} \tag{1.127}$$

where in the last equality we have used the definition of the face centroid, and where $\mathbf{S}_{f,P}$ is the face area vector pointing outward V_P. Notice that when using the Gauss theorem, we get the sum of surface integrals, hence we have to deal with expressions like

$$\sum_{f\in\partial V_P}\left(\int_f \mathbf{a}\cdot d\mathbf{S}\right) \approx \sum_{f\in\partial V_P}\mathbf{A}_f\cdot\mathbf{S}_{f,P}.$$

As stated before, the versus of the surface area vector $\mathbf{S}_f$ depends on its owner control volume: $\mathbf{S}_f$ points outwards from the control volume P only if f is *owned* by P, otherwise the *neighboring* face $\mathbf{S}_f$ points inwards. This fact must be taken into account when we consider $\mathbf{S}_{f,P}$ in the sum over the faces of the boundary, so the sum is split into sums over *owned* and *neighboring* faces:

$$\sum_{f\in\partial V_P}\mathbf{A}_f\cdot\mathbf{S}_{f,P} = \sum_{\substack{f\in\partial V_P\\ f\text{ owned by }P}}\mathbf{A}_f\cdot\mathbf{S}_f - \sum_{\substack{f\in\partial V_P\\ f\text{ neighb. of }P}}\mathbf{A}_f\cdot\mathbf{S}_f. \tag{1.128}$$

This is true for every summation over the faces, and in the rest of the text this split is automatically assumed.

For the **convective flux** discretization, the piece-wise linear approximation of the variables q and $\mathbf{u}$ at the face is used and we get that the integral on the face is approximated as follows:

$$\begin{aligned}\int_f (q\mathbf{u})\cdot d\mathbf{S} &\approx \int_f \left(\widetilde{Q}\widetilde{\mathbf{u}}\right)\cdot d\mathbf{S}\\ &\overset{(1.122)}{=} \int_f \Big\{\big[Q_f + (\mathbf{x}-\mathbf{x}_f)\cdot(\nabla\widetilde{Q})_f\big]\big[\mathbf{u}_f + (\mathbf{x}-\mathbf{x}_f)\cdot(\nabla\mathbf{u})_f\big]\Big\}\cdot d\mathbf{S}\\ &\approx \int_f (Q_f\mathbf{u}_f)\cdot d\mathbf{S} + \int_f Q_f\big[(\mathbf{x}-\mathbf{x}_f)\cdot(\nabla\mathbf{u})_f\big]\cdot d\mathbf{S} + \int_f\big[(\mathbf{x}-\mathbf{x}_f)\cdot(\nabla\widetilde{Q})_f\big]\mathbf{u}_f\cdot d\mathbf{S}\\ &= Q_f(\mathbf{u}_f\cdot\mathbf{S}_{f,P}) + Q_f(\nabla\mathbf{u})_f\cdot\int_f(\mathbf{x}-\mathbf{x}_f)d\mathbf{S} + \mathbf{u}_f(\nabla\widetilde{Q})_f\cdot\int_f(\mathbf{x}-\mathbf{x}_f)d\mathbf{S}\\ &\overset{(1.119)}{=} (\mathbf{u}_f\cdot\mathbf{S}_{f,P})Q_f,\end{aligned} \tag{1.129}$$

where, at the third passage, since we deal with a second order approximation, the second order term is neglected. The notation ϕ_f for the volumetric flux through the face f is now introduced:

$$\phi_f := \mathbf{u}_f\cdot\mathbf{S}_{f,P}, \tag{1.130}$$

then the entire convective term results as

$$\sum_{f\in\partial V_P}\left[\int_f (q\mathbf{u})\cdot d\mathbf{S}\right] \overset{(1.129)}{\approx} \sum_{f\in\partial V_P}(\mathbf{u}_f\cdot\mathbf{S}_{f,P})Q_f \overset{(1.130)}{=} \sum_{f\in\partial V_P}\phi_f Q_f. \tag{1.131}$$

It results that the divergence term discretization needs two schemes: the Gauss theorem (to treat the volume integral of the divergence term) and an interpolation scheme (like those presented in the next §1.3.2.2) to compute the value of the variable at the faces Q_f, whereas the computation of the volumetric flux ϕ_f will be described in §4.1. These two specifications, about schemes chosen, must be set in the `divSchemes` dictionary.

The **diffusive flux** is discretized in a similar way,hence, the approximation of the diffusive term is

$$\begin{aligned}\sum_{f\in\partial V_P}\left[\int_f (\Gamma\nabla q)\cdot d\mathbf{S}\right] &\overset{(1.127)}{\approx} \sum_{f\in\partial V_P}(\Gamma\nabla Q)_f\cdot\mathbf{S}_{f,P}\\ &= \sum_{f\in\partial V_P}\Gamma_f\mathbf{S}_{f,P}\cdot(\nabla Q)_f\,.\end{aligned} \tag{1.132}$$

From this expression we see that the Laplacian term discretization requires three schemes: the Gauss scheme (which is the only choice for the discretization of the volume integral of the Laplacian term), an interpolation scheme for the face values of the diffusion coefficient Γ_f, presented in the next paragraph §1.3.2.2, and finally a scheme for the surface-normal gradient term $\mathbf{S}_{f,P}\cdot(\nabla Q)_f$, which are introduced in a following paragraph §1.3.2.3. The three schemes chosen must be specified in the `laplacianSchemes`.

All the terms in the original equation (1.120) that cannot be expressed as temporal, convection or diffusion terms are then considered as sources. The **source term** is usually a function of space and time, of the solution and of other variables, so it might be difficult to treat. Before the actual discretization, the source term is linearized:

$$S(q)\approx S_u + S_p\widetilde{Q},$$

where both S_u and S_p might depend on the variable $\widetilde{Q}$. According with this, the volume integral of the source term is calculated as:

$$\int_{V_P} S(q)dV \approx (S_u + S_pQ_P)\,V_P. \tag{1.133}$$

In summary, by using the previous approximations (1.126), (1.131), (1.132) and (1.133) of the volume and surface integrals, Eq. (1.124) over the control volume V_P assumes the following semi-discrete expression:

$$\frac{\partial\,(Q_PV_P)}{\partial t} + \sum_{f\in\partial V_P}\phi_fQ_f - \sum_{f\in\partial V_P}\Gamma_f\mathbf{S}_{f,P}\cdot(\nabla Q)_f = (S_u + S_pQ_P)\,V_P \tag{1.134}$$

The solution of Eq. (1.134) in the control volume V_P requires the knowledge of the values Q_P, Q_f, Γ_f and $\mathbf{S}_{f,P}\cdot(\nabla Q)_f$, for each $f\in\partial V_P$. Remind that, due to the cell-centered co-located arrangement of the variables, only the values of the variables at the centroids of the control volumes are stored and used for the computations. Notice, instead, that

the convective and diffusive fluxes use the value of the variables at the faces. Because of this, an approximation of the value at the faces is computed in terms of the values at the cell-centroids, resulting in a sort of reconstruction over the entire control volume of the approximation function $\widetilde{Q}$. The values Q_P, Q_f and Γ_f are determined with interpolation schemes like those presented in the next paragraph §1.3.2.2, and the computation of $\mathbf{S}_{f,P} \cdot (\nabla Q)_f$ is done according with the surface-normal schemes described in a following paragraph §1.3.2.3. The time discretization schemes are introduced in §1.3.2.4.

1.3.2.2 Interpolation schemes

OpenFOAM has many interpolation schemes implemented, reported in Table 1.2 and divided into 4 categories: one category (the first row in the table) includes general schemes, whereas the other 3 categories (second, third and fourth rows in the table) involve the schemes to use in conjunction with the Gauss theorem. Among them, the convection-specific schemes calculate the interpolation based on the flux of the flow velocity. The specification of these schemes requires the name of the flux field on which the interpolation is based, that we called ϕ as most of the OpenFOAM applications do. It is not recommended that the user adopts any of the convection-specific schemes for generic field interpolations.

centered schemes:	`linear`
	`cubicCorrection`
	`midPoint`

Upwinded convection schemes:	`upwind`
	`linearUpwind`
	`skewLinear`

TVD schemes:	`limitedLinear`
	`vanLeer`
	`MUSCL`
	`limitedCubic`

NVD schemes:	`SFCD` `Gamma` ψ

Table 1.2: Interpolation schemes in OpenFOAM.

In the next paragraphs, we see three examples of interpolation schemes: the linear centered (`linear`), the classic upwind (`upwind`) and a scheme that mixes them (`limitedLinear`).

Central differencing. The Central Differencing (CD) scheme assumes a linear variation of $\widetilde{Q}$ between the centroids of two neighboring cells. Consider the one dimensional mesh shown in Figure 1.30. A simple linear interpolation is used to compute the face value:

$$Q_f = \theta_f Q_P + (1 - \theta_f) Q_N, \text{ where } \theta_f := \frac{\overline{fN}}{\overline{PN}}, \tag{1.135}$$

where the interpolation factor (weighting factor) is θ_f. The expression of the CD scheme complicates with more complex meshes, in multi-dimensional situations, and in case that the centroids P and N are not aligned with the face center f.

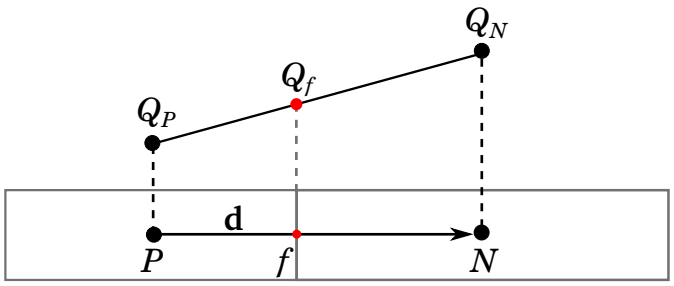


Figure 1.30: Assumption of linear variation of $\widetilde{Q}$ between two neighboring control volumes P and N. The face f is normal to the vector $\mathbf{d}_{PN}$.

Ferziger and Peric [89] showed that the CD scheme is second order accurate, hence it is consistent with the overall accuracy of the method. Even though the CFL condition is respected, the CD scheme can be unstable causing nonphysical oscillations in the solution for convection-dominated problems, thus violating the boundedness of the solution.

Upwind differencing. The Upwind Differencing (UD) scheme determines the value of Q_f looking at the direction of the flux, namely as:

$$Q_f = \begin{cases} Q_P & \text{if } \phi_f \geqslant 0, \\ Q_N & \text{if } \phi_f < 0. \end{cases} \tag{1.136}$$

As a clarifying example, consider the control volume P represented in Figure 1.31. It has four neighbor cells N, E, S, W, and the fluid flow is determined by the velocity field $\mathbf{u}$. In order to compute the values at the faces Q_f in the cell P, we need to know the sign of the volumetric flux ϕ_f at the faces. The faces owned by P are f_N and f_W, with the face-area vectors $\mathbf{S}_{fN}$ and $\mathbf{S}_{fW}$ that point outward. Reminding that, when determining the sign of ϕ_f, we must take the face-area vector owned by the neighbor cells with the minus sign; the upwind scheme is then applied as follows:

$$\begin{aligned} \phi_{fN} = \mathbf{S}_{fN} \cdot \mathbf{u}_{fN} \geqslant 0 \quad &\Longrightarrow \quad Q_{fN} = Q_P, \\ \phi_{fE} = (-\mathbf{S}_{fE}) \cdot \mathbf{u}_{fE} \geqslant 0 \quad &\Longrightarrow \quad Q_{fE} = Q_P, \\ \phi_{fS} = (-\mathbf{S}_{fS}) \cdot \mathbf{u}_{fS} < 0 \quad &\Longrightarrow \quad Q_{fS} = Q_S, \\ \phi_{fW} = \mathbf{S}_{fW} \cdot \mathbf{u}_{fW} < 0 \quad &\Longrightarrow \quad Q_{fW} = Q_W. \end{aligned}$$

The upwind scheme guarantees the boundedness of the solution, namely the stability. Despite this advantage, it is first order accurate then it violates the order of the discretization, also it produces high numerical diffusion/dissipation in the solution where large gradients exist, as stated also in §1.2.4.

Blended differencing. The Blended Differencing (BD) schemes try to preserve both boundedness and accuracy of the solution by using a linear combination of the CD and UD schemes as it follows:

$$Q_f = \theta Q_{f,CD} + (1-\theta) Q_{f,UD}, \qquad \text{with } 0 \leqslant \theta \leqslant 1, \tag{1.137}$$

where $Q_{f,CD}$ and $Q_{f,UD}$ are computed by Eq. (1.135) and Eq. (1.136) respectively. The blending factor θ prescribes how much numerical diffusion is introduced. For $\theta = 1$ the scheme reduces to CD, whereas for $\theta = 0$ the scheme is UD.

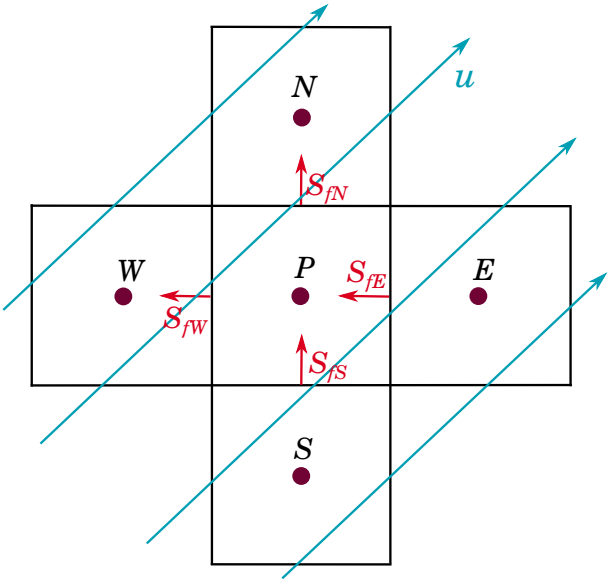


Figure 1.31: The control volume P and its adjacent cells. The *red* arrows represent the face-area vectors. The *blue* arrows represent the flow velocity.

As we saw in §1.2.3, diffusion may be considered as a raw cure to instability problems (those that born even if the CFL condition is respected). In such a context, the diffusive behavior introduced by the UD scheme reduces the generation of oscillations generated by the CD scheme. Peric [212] proposed a constant θ for all the faces of the mesh.

Eq. (1.137) is only one of the numerous blending schemes that try to find a good compromise between accuracy and boundedness.

1.3.2.3 Surface-normal gradient schemes

OpenFOAM offers several surface-normal gradient schemes to compute the quantity $\mathbf{S}_{f,P} \cdot (\nabla Q)_f$ in the diffusive flux (1.132); we report some of them in Table 1.3 and describe below with some details the `orthogonal`, `uncorrected` and `corrected` schemes. For these schemes, the choice of the user should take into account the mesh quality: it is orthogonal or not.

`orthogonal`	No non-orthogonal correction, 2^{nd} order accurate*, stable*
`uncorrected`	No non-orthogonal correction, 2^{nd} order accurate*, stable, but more diffusive than `corrected` and `limited`
`corrected`	Explicit non-orthogonal correction, 2^{nd} order accurate, bounded (depending on the mesh quality)
`limited` ψ	Limited non-orthogonal correction, 2^{nd} order accurate, bounded (depending on the mesh quality)

Table 1.3: `snGradSchemes`: some of the surface-normal gradient schemes available in OpenFOAM. The conditions * are valid for orthogonal hexaedral meshes with no grading.

In the case of orthogonal mesh, the vectors $\mathbf{d}$ and $\mathbf{S}_{f,P}$ are parallel, see Figure 1.32, and the scalar product may be computed through the following expression:

$$\mathbf{S}_{f,P} \cdot (\nabla Q)_f \approx |\mathbf{S}_{f,P}| \frac{Q_N - Q_P}{|\mathbf{d}_{PN}|}, \tag{1.138}$$

that corresponds to the `orthogonal` scheme.

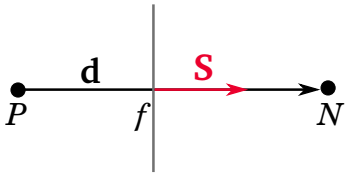
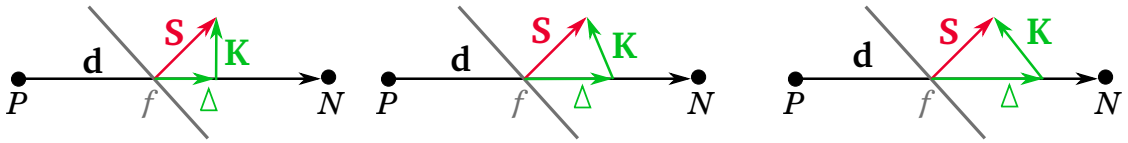


Figure 1.32: Face area vector $\mathbf{S}_{f,P}$ in an orthogonal mesh.

When the mesh is non orthogonal, the computation is split in two parts, the orthogonal contribution and the non-orthogonal correction, as follows:

$$\mathbf{S}_{f,P} \cdot (\nabla Q)_f = \underbrace{\boldsymbol{\Delta} \cdot (\nabla Q)_f}_{\text{orth. contribution}} + \underbrace{\mathbf{K} \cdot (\nabla Q)_f}_{\text{non-orth. correction}}, \quad \text{with } \mathbf{S}_{f,P} = \boldsymbol{\Delta} + \mathbf{K}, \tag{1.139}$$

where $\boldsymbol{\Delta}$ is parallel to the vector $\mathbf{d}$, therefore the first scalar product is approximated as in Eq. (1.138). The computation of the scalar product of the non-orthogonal correction depends on the chosen decomposition of $\mathbf{S}_f$, we see three of them. According with a *minimum correction* approach (also said *under-relaxed* approach), that aims to have the correction $\mathbf{K}$ as small as possible, the vector $\mathbf{K}$ is orthogonal to $\boldsymbol{\Delta}$, see Figure 1.33a. In the *orthogonal* approach, the length of $\mathbf{S}_{f,P}$ is reported along the direction PN, namely $|\boldsymbol{\Delta}| = |\mathbf{S}_{f,P}|$ as in Figure 1.33b. In the *over-relaxed* approach, that is represented in Figure 1.33c, the vector $\mathbf{K}$ is orthogonal to $\mathbf{S}_{f,P}$ and therefore the relevance of $\boldsymbol{\Delta}$ increases with the non-orthogonality.



(a) *Under-relaxed approach.* (b) *Orthogonal approach.* (c) *Over-relaxed approach.*

$$\boldsymbol{\Delta} = \frac{\mathbf{d} \cdot \mathbf{S}_{f,P}}{\mathbf{d} \cdot \mathbf{d}} \mathbf{d} \qquad \boldsymbol{\Delta} = \frac{|\mathbf{S}_{f,P}|}{|\mathbf{d}|} \mathbf{d} \qquad \boldsymbol{\Delta} = \frac{|\mathbf{S}_{f,P}|^2}{\mathbf{d} \cdot \mathbf{S}_{f,P}} \mathbf{d}$$

Figure 1.33: Three decompositions of the face area vector $\mathbf{S}_{f,P}$ in a non-orthogonal mesh.

The `uncorrected` scheme neglects the non-orthogonal correction and uses the under-relaxation approach, hence it should be used only for meshes with very low non-orthogonality:

$$\mathbf{S}_{f,P} \cdot (\nabla Q)_f \approx |\boldsymbol{\Delta}| \frac{Q_N - Q_P}{|\mathbf{d}_{PN}|}, \quad \text{with } |\boldsymbol{\Delta}| = \frac{1}{\cos\theta} |\mathbf{S}_{f,P}|,$$

where θ is the angle formed by $\mathbf{S}_{f,P}$ and $\boldsymbol{\Delta}$. The `corrected` scheme adopts the non-orthogonal correction and the under-relaxed approach, so the surface-normal gradient scheme results to be:

$$\mathbf{S}_{f,P} \cdot (\nabla Q)_f \approx |\boldsymbol{\Delta}| \frac{Q_N - Q_P}{|\mathbf{d}_{PN}|} + \mathbf{K} \cdot (\nabla Q)_f,$$
$$\text{with } |\boldsymbol{\Delta}| = \frac{1}{\cos\theta} |\mathbf{S}_{f,P}|, \quad \mathbf{K} = |\mathbf{S}_{f,P}| \left(\frac{\mathbf{S}_{f,P}}{|\mathbf{S}_{f,P}|} - \frac{1}{\cos\theta} \frac{\mathbf{d}_{PN}}{|\mathbf{d}_{PN}|} \right).$$

1.3.2.4 Time discretization

The original PDE, written in integral form as Eq.(1.123), has been discretized with respect to the space, then its approximation is the following equation:

$$\int_t^{t+\Delta t} \left[\frac{\partial \left(Q_P V_P\right)}{\partial t} + \sum_{f \in \partial V_P} \phi_f Q_f - \sum_{f \in \partial V_P} \Gamma_f \mathbf{S}_{f,P} \cdot (\nabla Q)_f \right] dt = \int_t^{t+\Delta t} \Big[\left(S_u + S_p Q_P\right) V_P \Big] dt \tag{1.140}$$

where the values at the faces are computed in terms of the centroid values. The time discretization is the last thing we need to obtain a full discretization.

The time discretization scheme to apply must be specified in the `ddtSchemes` dictionary. The time discretization schemes available in OpenFOAM are those listed in Table 1.4. There is an "off-centering" coefficient `ocCoeff` referred to `CrankNicholson` that is used for blending the CrankNicholson scheme with the Euler scheme. A coefficient `ocCoeff`= 1 corresponds to pure Crank-Nicholson method and `ocCoeff`= 0 corresponds to pure Euler scheme (we go deep in this in §4.2.1 and followings). The blending coefficient helps to improve stability in cases where pure Crank-Nicholson is unstable.

`Euler`	First order, bounded, implicit
`localEuler`	Local-time step, first order, bounded, implicit
`CrankNicholson ocCoeff`	Second order, bounded, implicit
`backward`	Second order, implicit
`steadyState`	Does not solve for time derivatives

Table 1.4: `ddtSchemes`: time discretization schemes available in OpenFOAM.

1.3.2.5 Example of `fvSchemes`

To run a simulation in OpenFOAM, the user must specify in the file case/system/fvSchemes the discretization schemes applied to the differential terms that appear in the equations to solve. We consider a particular example and we report the code of a `fvSchemes` file that shows a possible choice of discretization schemes to apply, for example the Code 1.1. The power of OpenFOAM is that the code is very readable, hence what reported could be almost intuitive. However, we pass through it in order to clarify what stated.

Suppose to have the following equation:

$$\frac{\partial q}{\partial t} + \nabla \cdot (q\mathbf{u}) - \nabla \cdot (\Gamma \nabla q) = S(q).$$

The equation is composed of three different differential terms that need individual discretization schemes to be specified in three dictionaries respectively: (*i*) `ddtSchemes`, (*ii*) `divSchemes` and (*iii*) `laplacianSchemes`. For each scheme, it is mandatory to write an instruction for the `default` choice. By stating `default none`, it is necessary to write explicitly a declaration for every term since no default scheme is defined. Otherwise, the user can define just a few terms and use the default for the rest. In the case of a system of equations, where several terms may have the same differential operator, the default instruction flexibility is useful.

```
18 ddtSchemes
19 {
```

```
20      default           Euler;
21  }
22
23  divSchemes
24  {
25      default           none;
26      div(phi,Q)        Gauss linear;
27  }
28
29  laplacianSchemes
30  {
31      default           Gauss linear orthogonal;
32  }
```

Listing 1.1: Example of code of `fvSchemes` file.

(*i*) The Code 1.1 states that the `Euler` scheme (reported in Table 1.4) is adopted for the time discretization.

In all the schemes for the spatial derivative terms (*ii*, *iii*), the Gauss theorem is used, and the keyword `Gauss` written in the Code 1.1 refers to this. The semi-discretization expression related to this keyword is

$$\frac{\partial\left(Q_P V_P\right)}{\partial t}+\sum_{f\in\partial V_P}\phi_f Q_f-\sum_{f\in\partial V_P}\Gamma_f\mathbf{S}_{f,P}\cdot(\nabla Q)_f=(S_u+S_pQ_P)V_P. \tag{1.141}$$

It remains to specify the schemes to determine Q_f, Γ_f, $\mathbf{S}_{f,P}\cdot(\nabla Q)_f$, and p_f.

(*ii*) The Code 1.1 indicates that the `linear` differencing scheme, defined in Eq. (1.135), should be used to compute Q_f. Remind that `phi` corresponds to the volumetric flux $\phi_f=\mathbf{u}_f\cdot\mathbf{S}_{f,P}$ defined in Eq. (1.130), and descends from the discretization of the convective flux, Eq. (1.129).

(*iii*) For the Laplacian term, we have two more specifications in addition to the string `Gauss` that indicates the use of Gauss theorem because we must choose two interpolations to compute Γ_f and $\mathbf{S}_{f,P}\cdot(\nabla Q)_f$. The first specification refers to the interpolation scheme to compute Γ_f (in this case it is `linear`, namely the centered scheme defined in Eq. (1.135)). The second specification denotes the surface-normal scheme that evaluates $\mathbf{S}_{f,P}\cdot(\nabla Q)_f$. In the Code 1.1 the `orthogonal` scheme is indicated (defined in Eq. (1.138)) that should be employed in case of orthogonal meshes.

The application of these schemes to Eq. (1.141) results in the following full discretization:

$$\begin{aligned}&\frac{Q_P^{n+1}-Q_P^n}{\Delta t}V_P+\sum_{f\in\partial V_P}\phi_f\left[\theta_f Q_P^{n+1}+(1-\theta_f)Q_N^{n+1}\right]+\\&-\sum_{f\in\partial V_P}\left[\theta_f\Gamma_P^{n+1}+(1-\theta_f)\Gamma_N^{n+1}\right]\frac{|\mathbf{S}_{f,P}|}{|\mathbf{d}_{PN}|}(Q_N^{n+1}-Q_P^{n+1})=\left(S_u+S_pQ_P^{n+1}\right)V_P.\end{aligned} \tag{1.142}$$

1.3.3 Algebraic system

For each control volume V_P, by summing the discretizations of the transient term, of all the flux terms and of the source term, the solver produces an *algebraic equation* which relates the variable under consideration at the centroid of V_P to the values at the centroids of the neighbor control volumes V_N. Doing this for each control volume, and assuming

that $\mathcal{N}$ is the number of the cells that constitute the computational domain, OpenFOAM finally assembles a set of $\mathcal{N}$ algebraic equations:

$$\mathbf{A}\mathbf{x} = \mathbf{b},$$

where the entries of $\mathbf{A}$ may depend on the unknowns. $\mathbf{A}$ is typically a sparse matrix (i.e., with many zero entries). We call A_P the diagonal coefficients and A_N the off-diagonal coefficients, so each equation may be written as follows:

$$A_P x_P + \sum_N A_N x_N = b_P, \tag{1.143}$$

This system is assembled and solved for each time step to determine the progress of the solution of the PDE.

In order to show how the discretization of a PDE reduces to an algebraic system, we rearrange the terms of the fully discretized Eq. (1.142) and identify the terms that build the system of Eqs. (1.143):

$$\underbrace{\left\{\frac{V_P}{\Delta t} - S_p + \sum_{f\in\partial V_P}\phi_f\theta_f + \sum_{f\in\partial V_P}\left[\theta_f\Gamma_P^{n+1} + (1-\theta_f)\Gamma_N^{n+1}\right]\frac{|\mathbf{S}_{f,P}|}{|\mathbf{d}_{PN}|}\right\}}_{A_P}\underbrace{Q_P^{n+1}}_{x_P}$$
$$+\sum_{f\in\partial V_P}\underbrace{\left\{\phi_f(1-\theta_f) - \left[\theta_f\Gamma_P^{n+1} + (1-\theta_f)\Gamma_N^{n+1}\right]\frac{|\mathbf{S}_{f,P}|}{|\mathbf{d}_{PN}|}\right\}}_{A_N}\underbrace{Q_N^{n+1}}_{x_N} = \underbrace{S_u + \frac{V_P}{\Delta t}Q_P^n}_{b_P}. \tag{1.144}$$

The coefficient A_P contains the contributes of all terms referred to Q_P^{n+1}: the temporal derivative, the linear part of the source term, the convective and diffusive terms. The coefficients A_N have the respective terms that correspond to the neighbor control volumes. The summation is equivalently done over the faces that belong to V_P or over the neighboring cells that share a face with V_P. The term b_P is constituted by all those terms that do not rely on the unknown at the new time step, hence the constant part of the source term, and the parts of the differential term discretization that depend on the variables at the previous time-step.

From the usage of the `Euler` scheme, we draw an important observation that is valid also for the more generic cases. Both the coefficients A_P and A_N may contain the term ϕ_f that accounts for the volumetric flux (defined in Eq. (1.130)). The volumetric flux depends on the velocity field $\mathbf{u}$. When the transported variable q is independent of velocity, for example, if it represents the density in the continuity equation $q = \rho$, Eq. (1.49a), the system is clearly linear. On the reverse, when the transported variable depends on velocity, as it happens for the momentum equation, Eq. (1.49b), where the momentum $q = \rho\mathbf{u}$ is transported by velocity, the system is quadratic and must be solved with proper strategies. In §4.1 we analyze classic solutions to these problems using linearization and a segregated approach.

The numerical algorithms that solve the system of equations fall in two main categories: *direct* and *iterative* methods. The **direct** algorithms compute the solution of the system in a finite number of arithmetic operations. This approach is generally more appropriate for small systems; in fact, the number of operations required to reach the solution often grows with the number of equations squared (which depends on the number $\mathcal{N}$ of cells that constitute the computational domain), making them prohibitively expensive for large

systems [195]. The **iterative** methods start with an initial guess of the solution and then improve the current approximation till the computed solution verifies a certain "tolerance". The iterative algorithms are less expensive than the direct methods, but they need the matrix to have some properties. (*i*) The sparsity of the matrix is a feature that makes the iterative solvers preserving this property very attractive because this allows to save computer memory. (*ii*) The convergence of the iterative methods must be ensured, for example if the matrix exhibits a *diagonal dominance*, namely if

$$|A_P| \geqslant \sum_N |A_N|, \quad \forall P = 1, \ldots, \mathcal{N},$$

with the inequality strictly verified for at least one row.

1.4 Lava flow application

In the previous sections, we presented the modeling and numerical contexts in which we developed our work, whereas in this section we introduce the final application of our effort: making simulations of the lava flows. In §1.4.1 we give a general overview of volcanic eruptions and give motivations on the importance of having accurate simulations for such phenomena. In §1.4.2 some more specific details are given about lava flows.

1.4.1 Volcanic eruptions

Volcanic eruptions represent the episodic or continuous surface discharge of magma from a storage region named the magmatic chamber. In the broadest sense, volcanic eruptions are either explosive or effusive, represented in Figure 1.34 (a-b) respectively. Eruption styles are often correlated with magma composition and with the energy involved. To some extent, this relationship reflects differences in tectonic settings, which also influences magmatic volatile content and magma supply rate. *Explosive eruptions* occur when the erupting magma is fragmented exiting the conduit. The reason is usually a high viscosity of the magma combined with high gas content. Dissolved gases cannot escape so easily because of the high viscosity (due to the magma composition), so pressure may build up until gas explosions blast rock and lava fragments into the air. The resulting fragments of the magma itself consist of small and large pieces: ash, lapilli, and bombs. If magma is sufficiently fluid not to fragment when gases expand approaching the surface vent, the magma can exit from the vent and flow downslope the topography. This is called *effusive eruption* and magma takes the name *lava*.

Volcanic eruptions are among the most destructive natural disasters that occur on our planet. Only in Italy there are at least ten active volcanoes (see Figure 1.35: Colli Albani, Campi Flegrei, Vesuvio, Ischia, Stromboli, Lipari, Vulcano, Etna, Pantelleria, and Isola Ferdinandea) which erupted in the last 10,000 years and two of them, Etna and Stromboli, have persistent activities, in fact, they erupt continuously or at most interrupt for brief periods of rest in the order of months or a few years.

Stromboli is considered one of the most active volcanoes in the world and is characterized by persistent explosive activity, just called *strombolian*, interrupted only by occasional effusive episodes of more intense activity, accompanied by *lava flows*, as recently occurred in 1975, 1985, 2003, 2007 and 2014. Etna is the largest volcano in Europe and one of the most active volcanoes in the world. Etna is considered an effusive volcano predominantly because the emission of lava mainly characterizes it. Lava flows can cause damage but

(a) (b)

Figure 1.34: (a) Explosive eruption of Mount Fuego (Guatemala), 2015. (b) Effusive eruption of Mount Kilauea (Hawaii), May 2018.

do not represent a direct threat to the lives of the 900,000 people living in potentially hazardous areas. However, recently it has been observed, especially from the late 70s, a strong increase of explosive eruptive episodes, especially at the summit craters.

Although volcanic eruptions do not often cause large humanitarian disasters, they have the potential to induce extreme ones [76, 203]. For example, volcanically induced mudflows caused 21800 deaths in Colombia in 1985, and CO_2 poisoning due to a crater lake's overturn caused 1746 deaths in Cameroon in 1986. Both events lasted minutes to hours and impacted a single catchment but caused near-complete destruction within the impacted catchment. Even large-scale evacuations for extended periods should be considered as disasters, even if not many people die, as, for example, in the cases that follow. The eruption of the Merapi volcano, Indonesia, in late 2010, killed more than 300 people, posed significant challenges for evacuations, and resulted in a peak number of almost 400000 internally displaced people [189]. Similarly, Indonesia's Mt. Sinabung eruptions, February 2014, caused 16 deaths and continuous evacuations of 30,000 residents [36]. More, the eruption of Fuego volcano, Guatemala, on 3rd June 2018, had tragic outcomes when pyroclastic flows inundated an entire village, and the eruption has prompted evacuations of around 12000 people [176]. Finally, the 2017–19 activity at Mt. Agung in Bali, Indonesia, saw the delay between intense unrest and eruption causing considerable challenges to emergency responders, local and national governmental agencies, and the population of Bali near the volcano, including over 140,000 evacuees [250]. Some explosive volcanic eruptions impacted vast populations, for example, 1 million people in 1991 in the Philippines, plus around 300000 in Nicaragua (1992), Ecuador (2006), and Indonesia (1982) [111] and ma-

Figure 1.35: Volcanoes in Italy.

jor historical eruptions, such as Laki 1783 (Iceland) [256], or Tambora 1815 (Indonesia) [249], had a regional or even global climatic and economic impact [75].

Like any volcanic eruption, an effusive event cannot be stopped (see, for example, Figure 1.36 showing the destructive impact on the civilian infrastructures and houses of the Kilauea eruption in 2018). However, effusive flows are relatively slow in propagation because lava flow fronts tend to advance at a few hundred meters per hour most of the time. Hence there is time to respond to an advancing lava flow once the event is underway. We may thus argue that the effusive-event scenario is relatively easy to prepare for, being there time to issue a call to scientific and civil protection responders to set up event scenarios and response plans in almost real-time. So, if volcanic eruptions are a major concern in terms of casualties, evacuation needs, and damage, effusive eruptions also require an adequate humanitarian response owing to the hazard they pose to human populations in their catchments [19]. Anyway, in addition to the needs, costs, and logistics of evacuation, intervention, replacement, and relocation for these populations, we must consider the need for mental health care, measures against the collapse of social structures, and the maintenance of law and order. Table 1.5 reports some of the major effusive events that happened after the Second World War, which involve 12 volcanoes for 38 eruptions.

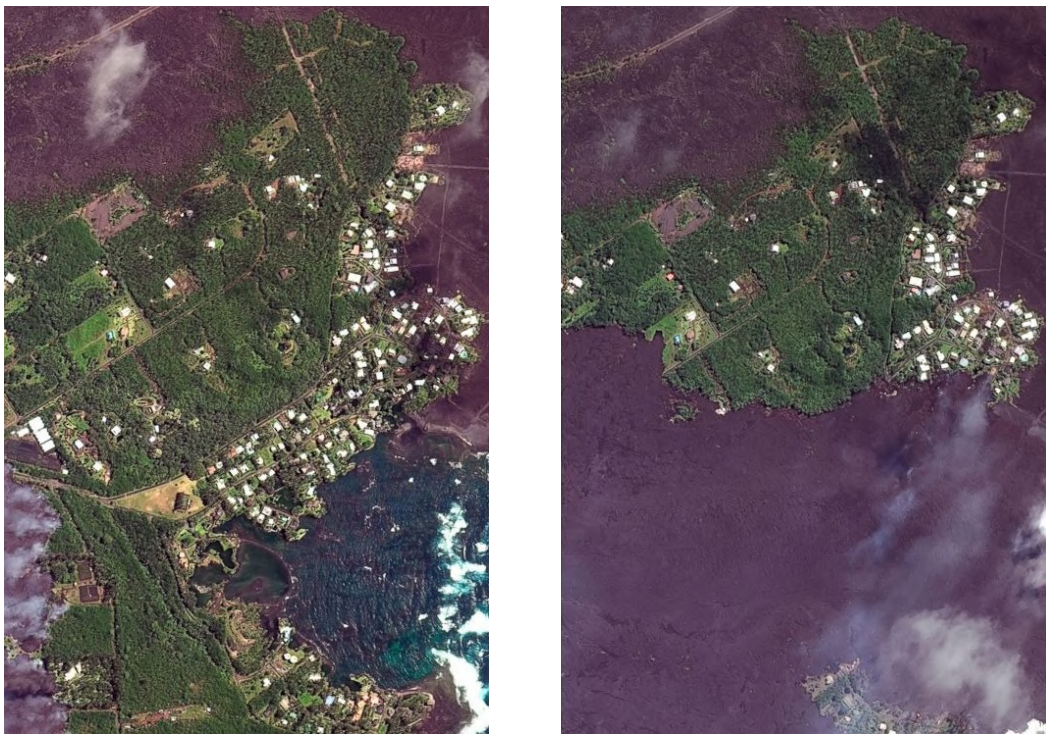


Figure 1.36: Satellite images of near Kapoho Bay in The Big Island (Hawaii), before and after Kilauea lava flow propagation, June 2018.

Even though lava may be slow and take some time before getting in touch with a civilian community, evacuation is necessary for those communities along the path of the lava flow [120]. Lava burns and buries everything in its path [19, 120], hence also a post-event plan for the replacement of infrastructures, society, and community is required if human interventions have not succeeded. Human interventions may consist of using ditches and barriers to deflect the lava flow, and also bombs, explosives, and/or water are employed. All such measures require preparation, planning, and allocation of money and resources. For example, the deflection barriers constructed to protect vulnerable

infrastructure on the south flank of Mount Etna during the 1983 eruption required the construction of 10 km of service roads and delivery of 750000 m^3 of rock to the construction site over a period of 50 days at 13 h/day [47]. Shipping 103 m^3 of rock to the site each hour required 20 trucks per hour, with the total cost being 3678 million Italian lire or 1.9 million Euros.

Table 1.5: Some of the major effusive eruptions of the last 70 years, after the Second World War, are reported. Consult [75, 259] for the damages and impacts caused by the eruptions on the civil population.

Volcano	Eruption
Eldfell (Iceland)	1973
Etna (Italy)	1971, 1981, 1983, 1991–1993, 2001, 2002–2003
Fogo (Cape Verde)	1951, 1995, 2014–2015
Izu-Oshima (Japan)	1986
Karthala (Comoros)	1977
Kilauea (USA)	1955, 1960, 1983–1991, 2018
La Palme (Spain)	1971
Mauna Loa (USA)	1950, 1984
Miyakejima (Japan)	1962, 1983
Mt. Cameroon (Cameroon)	1959, 1982, 1999, 2000
Nyiragongo (Democratic Republic of Congo)	1977, 2002
Piton de la Fournaise (France)	1977–1978, 1980, 1986, 1998, 2001, 2002, 2004, 2005, 2007, 2018

Oversimplifying modern volcanology, we can distinguish three main approaches through which volcanologists study volcanic eruptions:

1. monitoring of volcanoes (for example, observing the long-term and short-term seismograph surveys, variations of temperature registered by thermographic cameras, changes in the gravitational field, the composition of the gases in the air) to know the current status of the volcanic system because some phenomena are assessed together as signs (precursors) of an imminent eruption; based on the information provided by the monitoring instruments, it is decided whether justify an evacuation;

2. reconstructing the eruptive history using stratigraphic and analytical studies on the volcanic products of past eruptions (namely on the scoria composing the cone, the solidified lava, lapilli, and ashes) and studies of the cameras videos (if present on the land during the eruptive event) and of the satellite pictures as far as the most recent events concern;

3. modeling the volcanic processes using physical and mathematical models, statistical models, and numeric simulations.

From the integration of these approaches, volcanologists derive the eruptive scenarios underlying the emergency and mitigation plans.

In this thesis, we deal with the modeling approach and the production of numerical simulations. The first objective of the physical-mathematical modeling is improving the

knowledge of the physical processes governing the dynamics of such events. The second goal is to contribute to the quantification of the hazard and risks associated with volcanic eruptions. In this work, we are mainly interested in the first objective, particularly in modeling the geophysical process of ***lava flows*** that characterize the effusive events.

1.4.2 Lava flows

Lava is a viscous fluid, hence the Navier-Stokes Eqs. (1.37) are the mathematical tool to describe its behavior. In a first approximation, lava flows are gravity currents, whereby topography has a major role in determining their evolution. Secondly, the chosen rheological and viscosity models (introduced in §1.1.5.7) make another difference in determining the dynamics. The third major factor characterizing lava flows is the temperature because it has an enormous impact on viscosity. Indeed, lava viscosity is closely related both to temperature, its chemical composition (for example, basaltic, andesitic, and rhyolitic lava distinguish on the percentage of silica that contain and that leads to different eruptive behaviors), and other factors which might be, for example, the crystal fraction (as in the celebrated Krieger-Dougherty equation [154]) or the number of bubbles (for a magmatic fluid). A viscosity model accounting temperature is, for example, the Costa and Macedonio [55] lava model. In addition, during the cooling process, the lava may produce rafts of solidified material on the free surface (that move with the flow) or create a superficial crust that insulates lava from the atmosphere and reduces its cooling. Finally, the last fundamental factors influencing the lava flow dynamics in terms of flow extent are the effusion rate and the erupted volume. Most of these factors are related to each other and contribute to determine the final lava flow extent and emplacement.

In a pioneering work [268], the author studied the Mount Etna eruption in 1971 and set the basis for understanding the relationship between the effusion rate, lava emplacement, and flow field type. From such work descended that viscosity is a secondary factor in the relationship governing the interplay between the flow length, heat loss, and effusion rate [117]. The author observed that higher effusion rates corresponded to long and straightforward flows emplacements, whereas lower effusion rates produced multiple and short flows that pile up around a vent and build compound flow fields. Also, the author noticed the counterintuitive fact that relatively low viscosity flows tend to propagate less than the somewhat more viscous flows. The author conjectured that this tendency was caused by some factors, other than viscosity, that control the lengths of the lava flow. For example, flows with relatively low viscosity are thinner than the other and so are affected by a relatively higher rate of heat loss per unit volume, this could be one reason behind the shorter extent. From these observations the complexities involved in determining how far a flow can extend emerge. The effusion rate, rheology, heat loss, and eruption duration all play roles, and all are subject to complex feedbacks with each other. Heat loss, for example, determine an increase of viscosity (that is temperature-dependent), which leads to a velocity decrease, eventually limiting a flow's ability to move.

Under particular conditions of effusion rate, temperature, and chemical-physical composition of the lava, the superficial crust that might develop during the cooling process behaves like a tube [110], see Figure 1.37a, allowing the lava to flow under it faster (because the crust keeps the lava hotter) and letting it extend further. In other conditions, the crust may produce inflation [222], a phenomenon for which the lava beneath lifts the crust upwards (increasing the lava flow thickness even of tens of meters), see Figure 1.37b. When the viscosity is very high, lava-domes arise, that is, mound-like structures that form

directly over the volcano's vent by the build-up of lava [90], see Figure 1.37c.

(a) *A peek through a "skylight" into the interior of an active lava tube, Hawaii, US.* From: U.S. Geological Survey, Ph: J. Judd.

(b) *Inflation process at Kilauea, Hawaii, US.* From: Volcano Discovery, Ph: Tom Pfeiffer.

(c) *Volcanic domes in the crater of Mount Saint Helens, southwestern Washington, US.* From: U.S. Geological Survey, Ph: Willie Scott.

(d) *Fully molten lava flow, Mount Fagradalsfjall in 2021, Iceland.* From: Phys.org.

Figure 1.37: Lava flows which different behavior depends on the different physico-chemical and tectonic conditions.

Despite the high complexity related to lava flows, in this thesis, we consider only the case of completely molten lava, see Figure 1.37d. So, we use the incompressible Navier-Stokes Eqs. (1.49) adopting both a Newtonian rheological model for viscosity (Eq. (1.47)) because completely molten lava shows this rheology, as stated in [107], and a Bingham plastic rheological model (Eqs. (1.51,1.52)).

Chapter 2

Mathematical models

This chapter presents two different models that describe the motion of a free surface, incompressible, Bingham plastic fluid in a laminar regime. We also consider that the fluid is warmer than the surrounding environment and that it cools down because of heat exchanges with the environment and ground. In addition, the fluid viscosity may be affected by the temperature changes. We derive the two models from the incompressible Navier–Stokes Eqs. (1.49), and they differ in the number of spatial dimensions explicitly involved.

A depth-averaged model is derived in section §2.1 by considering a shallow water approximation. Shallow water equations are widely used to simulate those geophysical flows for which the flow horizontal length scale is much greater than the vertical one. Inspired by the example of lava flows, we derive a modified shallow water model that also presents an additional transport equation for a scalar quantity. In the first instance, we consider a situation with only the advection of such scalar quantity. After, we assimilate that scalar quantity to temperature and model the heat exchanges between the fluid and the environment. Having the temperature description, we use it in a temperature-dependent viscosity model. The derivation of the governing equations, from depth-averaging the incompressible Navier-Stokes equations, is presented. Because of the depth-averaging procedure, the problem reduces by one dimension, and this model comes out to be 2D. The assumption of constant vertical profiles is a common hypothesis of the classic shallow-water approximation; in our modified model, we relax such assumption for some of the model variables allowing the presence of vertical profiles. It also follows that some appropriate shape coefficients are introduced because of the non-linearity of the flux terms.

Depth-averaged models have the drawback to fixing a priori the vertical distribution of the variables, both in the classical and our model. For this reason, we also consider a full 3D model and will dedicate the section §2.2 to its derivation. The 3D model presents the simultaneous description of the dynamics of both the free surface fluid of our interest and of the air around it (which respects the incompressibility condition as well), leading to a *multiphase* model. Furthermore, because of the approach we choose, an additional equation for transporting the phases joins the system. Finally, the resulting governing equation system also has an equation for the thermal energy that describes the heat loss from the warm fluid to the surrounding environment and soil, similar to what happens in the depth-averaged case.

In section §2.3, we review alternative ways to treat the multiphase flows from the one we adopted for the 3D model, namely the VOF method; furthermore, we also present the variety of possible approaches to the free surface fluids modeling different from the depth-

averaged and the 3D models, with a particular focus on the codes currently available for the lava flow simulations.

2.1 Depth-averaged model

Shallow water equations are a depth-averaged system of partial differential equations describing the dynamics of inviscid and constant-density fluids for which the flow horizontal length scale is much greater than the vertical one. In the past two centuries, they have been widely used to simulate a multitude of geophysical flows. In the volcanologic field, for example, they are applied with success to the simulation of pyroclastic density currents [244], lava flows [55, 145], and lahars [221]. In the meteorological field, they are used to describe the horizontal structure of the atmosphere with an additional term modeling Coriolis forces [11, 273]; in the study of ocean circulation, they are employed to produce global, realistic tidal models [179], and they accurately describe the propagation of a tsunami until waves approach the shore [85, 182]. In work about the dunes' evolution [132], a modified depth-averaged approach was used to model the transport of sediment.

The classical shallow water equations, first introduced by De Saint Venant in 1864 and Boussinesq in 1872, are based on several assumptions, among which the fact that the vertical pressure distribution is hydrostatic, the vertical component of the velocity can be neglected, and that the horizontal velocity field can be considered constant with depth. In the past, several modifications have been proposed to weaken this last assumption and to obtain a better model for flows where vertical shear is essential, resulting in a modified momentum equation where a multiplying coefficient (named *shape* or *Boussinesq factor*) is introduced in the advective flux term [24]. Its magnitude relates the mean square velocity to the square of the mean velocity and reflects the shear in the profile of the horizontal fluid velocity, and may depend on factors such as the Reynolds number (defined in Eq. (1.54)) or the boundary roughness. It is well known that these modifications do not change the hyperbolic nature of the equations, but in some cases, they can significantly impact the front features and propagation [128].

From the modeling point of view, in this contribution, we go a step further, considering viscosity and enriching the classical model with the parameter describing the vertical variation of the velocity field (the Boussinesq coefficient), see §2.1.1, and with an additional transport equation for a scalar quantity varying horizontally and with a non-constant vertical profile, see §2.1.4. This is a pretty standard feature in environmental science where. For example, this quantity may represent the concentration of sediment-laden flows [35], where a high-concentration basal layer may affect the transport dynamics or pollutants [169]. In geophysics and volcanological applications, it may represent the fraction of solid particles in pyroclastic flows [67] or the temperature in lava flows [55], where a thermal boundary layer develops when fluids flow over a solid, or it may represent both the temperature and the salinity of water in oceanology [274]. Even for this quantity, as done for the velocity field, a shape coefficient describing the vertical profile is introduced, and the resulting depth-averaged transport equation is derived. We also present a modified version of such transport equation that is designed explicitly for temperature, describing the thermal exchanges between the fluid and the surrounding environment, see §2.1.5. In addition, we also consider the case where density depends on the depth-averaged temperature; therefore, density varies horizontally, further relaxing the assumptions of the classical shallow water equations. Moreover, the equations have terms that describe the emission of new fluid into the system from a source point. After deriving the depth-averaged model,

we study the hyperbolicity of the system (see Definition 1.4) highlighting the role of the shape parameters, see §2.1.6.

Other ways to derive the shallow water equations pass through the use of variational principles or asymptotic expansions. In Appendix B we provide some background notions about variational principles and calculus of variations which are used to derive both the classic Saint-Venant equations and a modified version of them where correction terms related to the topography irregularities are present (by taking inspiration from the works of Clamond and Dutykh [45] and Clamond and Dutykh [46]). Instead, the derivation based on the asymptotic expansions uses the solutions of the Cauchy momentum equations in the shallow water scaling and in the neighbourhood of steady solutions. Such a method was first introduced for the case of Newtonian fluids [29], then in the more complex case of arbitrary topography [25], and eventually even in the case of power-law fluids and Bingham fluids [86, 87].

Assumptions

We present the derivation of a depth-averaged model for the dynamics of an incompressible, laminar, viscous, and homogeneous fluid over variable topography. The model also considers the transport of an additional quantity with the vertical distribution. Even though the depth-averaged equations are commonly derived directly from the two principles of the conservation of the mass and momentum, here we obtain them by depth-averaging the general 3D equations. In the Eulerian framework, the equations are derived by depth-averaging the Navier-Stokes equations assuming that the horizontal length scale is much greater than the vertical length scale. Since we are interested in incompressible flows even with non-constant density, we refer to the incompressible formulation of the Navier-Stokes Eqs. expressed in (1.49). This choice is because in many geophysical flows, like in oceans, lava flows, or debris flows, fluids are incompressible but with variable density, for example, as a result of spatial and temporal variations in temperature (for instance, due to thermal exchanges and cooling of a lava flow), in the concentration of sediments (due to settling or erosion processes) or in the concentration of salinity.

We assume that the horizontal length scale is much greater than the vertical length scale and that the vertical dynamics is negligible compared to horizontal effects (that is, the velocity vector is $\mathbf{u} = (u, v, w)$ with $w = 0$); by integrating over the flow thickness, it is possible to derive a simpler 2D transient model describing the fluid dynamics. Here, we write this simplified system of equations by adopting a horizontal Cartesian system of coordinates such that the topography (assumed not varying with time) expresses by the function $B(x, y)$ and the two significant components of velocity, u and v, are defined as the components along the x and y axes respectively. The plane defined by the x and y axes is orthogonal to the z axis which is parallel to the gravitational acceleration ($\mathbf{g} = (0, 0, g)$). We introduce two additional variables describing the system: $h(x, y, t)$, which denotes the fluid thickness above the ground, and $\mathcal{T}(x, y, z, t)$, which is the field of an additional transported intensive property, i.e., a local physical property of a system that does not depend on the system size or the amount of material in the system. For clarity, from now on, we assume that the transported quantity will represent the temperature of the flow, as it would be for a lava flow.

Following these assumptions, we introduce the notation $\mathbf{U}(x, y, t)$ for the z-averaged

horizontal velocity vector, whose two components U and V are given by

$$U(x,y,t) = \frac{1}{h}\int_B^{B+h} u(x,y,z,t)\,dz, \qquad V(x,y,t) = \frac{1}{h}\int_B^{B+h} v(x,y,z,t)\,dz. \tag{2.1}$$

Similarly, we introduce the notation T for the depth-averaged temperature

$$T(x,y,t) = \frac{1}{h}\int_B^{B+h} \mathcal{T}(x,y,z,t)\,dz. \tag{2.2}$$

Density depends on the temperature in various situations, and we assume a linear dependence as follows density,

$$\rho(x,y,z,t) = \rho_0 + m\mathcal{T}(x,y,z,t). \tag{2.3}$$

that is precisely the case of some of the materials (wax and silicone oils) investigated in the examples presented in §3.4. For the applications we are interested in (for example, lava flows or the laboratory analog experiments simulated in this work), the vertical variations of density are minor with respect to the horizontal ones, for such reason, we will consider only horizontal variations of the depth-averaged density. By observing that also the depth-averaged density depends linearly on the depth-averaged temperature

$$\bar{\rho} := \frac{1}{h}\int_B^{B+h} \rho(z)\,dz = \frac{1}{h}\int_B^{B+h} \Big[\rho_0 + m\mathcal{T}(z)\Big]\,dz = \rho_0 + mT,$$

then, in the following, we will use for the depth-averaged density, without ambiguity, the notation ρ. Since we mainly speak about temperature, we have $m < 0$ because the density decreases when the temperature grows. In other situations, where the transported quantity has a different physical meaning (e.g., particle concentration), the sign of m could be different. In section §2.1.7 we will deal with a preliminary treatment of the more general situation where even the density assumes a vertical profile.

2.1.1 Velocity profile

In the derivation of the classical shallow-water model, from the assumption of inviscid fluid, it descends that the velocity does not depend on the vertical coordinate, i.e., $\mathbf{u} = \mathbf{u}(\mathbf{x},t)$. With this assumption, the depth-averaged velocity $\mathbf{U}$ corresponds to the local velocity. Instead, for a free surface laminar viscous flow, moving in one direction, a vertical velocity profile develops, and three conditions must be satisfied: (i) null velocity at the bottom, (ii) null traction between air and fluid, and (iii) maximum velocity at the free surface.

When considering the motion of a Newtonian fully developed laminar viscous flow, the balance of friction and gravitational force leads to a parabolic velocity profile, as represented in Figure 2.1 and shown in [73], similarly to a Poiseuille flow:

$$\mathbf{u}(\mathbf{x},z,t) = \mathbf{a}(z-B)^2 + \mathbf{b}(z-B) + \mathbf{c}, \tag{2.4}$$

with $\mathbf{a} = \mathbf{a}(\mathbf{x},t)$, $\mathbf{b} = \mathbf{b}(\mathbf{x},t)$, $\mathbf{c} = \mathbf{c}(\mathbf{x},t)$. The parabolic profile is assumed for both the components of velocity $\mathbf{u} = (u,v)$. By imposing the previous conditions to the expression of $\mathbf{u}(\mathbf{x},z,t)$ in Eq. (2.4), we completely determine its expression:

(i) $\mathbf{u}(\mathbf{x},B,t) = 0 \iff \mathbf{c}(\mathbf{x},t) = 0,\ \forall\mathbf{x},\ \forall t;$

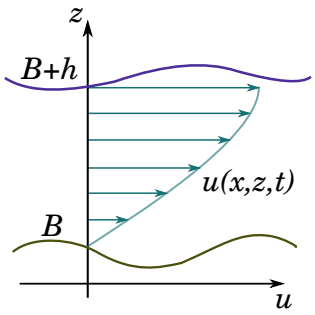


Figure 2.1: Parabolic velocity profile.

(ii) $\partial_z \mathbf{u}(\mathbf{x}, B+h, t) = 0 \iff \mathbf{b}(\mathbf{x}, t) = -2\,h\,\mathbf{a}(\mathbf{x}, t),\ \forall \mathbf{x},\ \forall t;$

(iii) $\mathbf{u}(\mathbf{x}, B+h, t) = \mathbf{a}(\mathbf{x}, t)h^2 + \mathbf{b}(\mathbf{x}, t)h \iff \mathbf{a}(\mathbf{x}, t) = -\dfrac{1}{h^2}\mathbf{u}(\mathbf{x}, B+h, t),\ \forall \mathbf{x},\ \forall t.$

For this parabolic profile, from the definition of $\mathbf{U}$ in Eq. (2.1), we find a relation between the depth-averaged velocity and the velocity at the free surface

$$\begin{aligned}
\mathbf{U} &\overset{(2.1)}{=} \frac{1}{h}\int_B^{B+h} \mathbf{u}(z)dz \\
&= \frac{1}{h}\int_B^{B+h} -\frac{\mathbf{u}(B+h)}{h^2}\Big[(z-B)^2 - 2h(z-B)\Big]dz \\
&= -\frac{\mathbf{u}(B+h)}{h^3}\left[\frac{(z-B)^3}{3} - h(z-B)^2\right]_B^{B+h} \\
&= \frac{2}{3}\mathbf{u}(B+h)
\end{aligned}$$

and, finally, by applying (i)–(iii) and the previous relation, we are able to express $\mathbf{u}$ in terms of $\mathbf{U}$ as

$$\mathbf{u}(\mathbf{x}, z, t) = \frac{3}{2}\left\{1 - \left[\frac{z - \big(B(\mathbf{x}) + h(\mathbf{x}, t)\big)}{h(\mathbf{x}, t)}\right]^2\right\}\mathbf{U}(\mathbf{x}, t). \tag{2.5}$$

To summarize, even though the shallow water model solves for the depth-averaged velocity $\mathbf{U}$, by using the assumptions of parabolic profile $(i-iii)$, we succeed to obtain an useful explicit expression of the velocity vertical distribution $\mathbf{u}(z)$ in terms of $\mathbf{U}$, the free-surface high $B+h$ and the fluid thickness h. In the following, we assume the parabolic profile given by Eq. (2.5).

Continuity equation. We integrate the equation of the conservation of the mass (1.49a) over the flow thickness

$$\int_B^{B+h}\left[\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\mathbf{u})\right]dz = \frac{\partial}{\partial t}\int_B^{B+h}\rho\,dz + \frac{\partial}{\partial x}\int_B^{B+h}\big(\rho u\big)dz + \frac{\partial}{\partial y}\int_B^{B+h}\big(\rho v\big)dz,$$

and, reminding that the density ρ does not depends on the vertical coordinate, we obtain the *depth-averaged continuity equation*:

$$\frac{\partial(\rho h)}{\partial t} + \nabla\cdot(\rho h\mathbf{U}) = 0. \tag{2.6}$$

One observes that, despite the presence of a velocity profile, the depth-averaged continuity equation does not present any formal difference from the classical expression of Eq. (1.49a).

When one wants to model also the release of new material into the system, the source term ρR must be added on the right hand side of the equation, where R represents the volumetric rate of fluid per unit area

$$\frac{\partial(\rho h)}{\partial t} + \nabla \cdot (\rho h \mathbf{U}) = \rho R. \tag{2.7}$$

Hyperbolic terms of the momentum equation. When the transient term of the momentum equation (1.49b) is integrated over the flow thickness, because of the assumption that ρ does not depend on the flow depth, we have:

$$\int_B^{B+h} \frac{\partial(\rho \mathbf{u}(z))}{\partial t} dz = \frac{\partial}{\partial t} \int_B^{B+h} \rho \mathbf{u}(z)\, dz = \frac{\partial(\rho h \mathbf{U})}{\partial t}. \tag{2.8}$$

Again, one notes that the presence of a velocity vertical profile does not modify the transient term expression.

In the derivation of the advective flux term of the *depth-averaged momentum equation*, we consider, for the sake of simplicity, the 1D model. By substituting the expression for the velocity profile in the advective term and by integrating it, one obtains

$$\begin{aligned}
\int_B^{B+h} \rho\, u^2(z)\, dz &\overset{(2.5)}{=} \int_B^{B+h} \rho \frac{9}{4} \left\{ 1 - \left[\frac{z - (B+h)}{h} \right]^2 \right\}^2 U^2 dz \\
&= \int_B^{B+h} \frac{9}{4} \rho U^2 \left\{ 1 - 2\frac{\left[z - (B+h)\right]^2}{h^2} + \frac{\left[z - (B+h)\right]^4}{h^4} \right\} dz \\
&= \frac{9}{4} \rho U^2 \left\{ z - \frac{2}{3}\frac{\left[z - (B+h)\right]^3}{h^2} + \frac{1}{5}\frac{\left[z - (B+h)\right]^5}{h^4} \right\}_B^{B+h} \\
&= \frac{9}{4} \rho U^2 \left\{ h + \frac{2}{3}\frac{[-h]^3}{h^2} - \frac{1}{5}\frac{[-h]^5}{h^4} \right\} \\
&= \frac{9}{4} \rho h U^2 \left\{ 1 - \frac{2}{3} + \frac{1}{5} \right\}
\end{aligned}$$

so that one finds:

$$\int_B^{B+h} \rho\, u^2(z)\, dz = \beta_u \rho\, h\, U^2, \tag{2.9}$$

where $\beta_u = \frac{6}{5}$. This coefficient is often termed in literature as *Boussinesq momentum coefficient* or *shape factor* or *corrector factor*, and its magnitude reflects the shear in the profile of the horizontal fluid velocity. Different velocity profiles, arising, for example, when non-Newtonian viscosity applies, will result in different values of β_u. Even though it is frequently set equal to unity (which holds only when velocity is constant over the flow thickness), it is well known that this coefficient may have a significant effect on the dynamics of the flow when a complete sheared flow is expected [128].

In the advective flux term, because of its non-linearity with respect to velocity, we finally see a difference from the classical shallow water equations in the case that a non-constant vertical distribution is adopted for the velocity. It is worth to add that the value

of β_u is $\frac{6}{5}$ even in the case that one considers the parabolic profile represented in Figure 2.2, where the maximum velocity is at the average depth and it is null both on the bottom and at the surface. This case may represent, several situations such as the fluid motion in a pipe, or the fluid motion between two steady plates, or the situation of a "lava tube" in which case the lava surface is completely solidified and does not move.

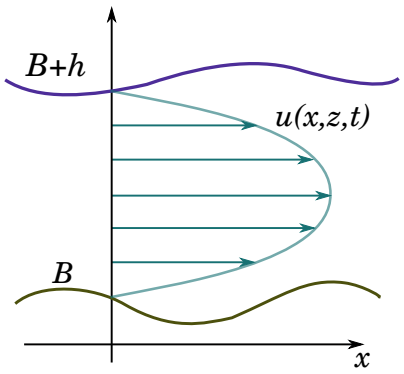


Figure 2.2: A different kind of parabolic profile for velocity.

2.1.2 Pressure and viscosity terms

Let assume now a hydrostatic profile for the pressure p, i.e.

$$\frac{\partial p(\mathbf{x}, z, t)}{\partial z} = \rho(\mathbf{x}, t) g.$$

Integrating p from the height z to the free surface $B + h$, and fixing at 0 the pressure value at the free surface, we obtain the expression of hydrostatic pressure at height z

$$p(\mathbf{x}, z, t) = \int_z^{B+h} \frac{\partial p(\mathbf{x}, \zeta, t)}{\partial \zeta} d\zeta = \rho(\mathbf{x}, t) g \Big[\Big(B(\mathbf{x}) + h(\mathbf{x}, t) \Big) - z \Big]$$

and, deriving with respect to x, we have

$$\frac{\partial p(\mathbf{x}, z, t)}{\partial x} = g \frac{\partial}{\partial x} \Big[\rho(\mathbf{x}, t) \Big(B(\mathbf{x}) + h(\mathbf{x}, t) \Big) \Big] - g z \frac{\partial \rho(\mathbf{x}, t)}{\partial x}$$

When this term is integrated with respect to z over the flow thickness, i.e. it is integrated from $z = B$ to $B+h$, we obtain the desired term of the depth-averaged equation accounting for pressure gradient for the x component, that is

$$\int_B^{B+h} \frac{\partial p}{\partial x} dz = gh \frac{\partial}{\partial x} \left[\rho \left(B + h \right) \right] - g \frac{\partial \rho}{\partial x} \left[\frac{z^2}{2} \right]_B^{B+h} = \rho g h \left(\frac{\partial h}{\partial x} + \frac{\partial B}{\partial x} \right) + \frac{1}{2} g h^2 \frac{\partial \rho}{\partial x}.$$

By rearranging the derivatives, the depth-integrated pressure gradient reduces to

$$\int_B^{B+h} \frac{\partial p(z)}{\partial x} dz = \frac{\partial}{\partial x} \left(\frac{1}{2} \rho g h^2 \right) + \rho g h \frac{\partial B}{\partial x}, \tag{2.10}$$

which is the term usually found in *depth-averaged momentum equations* in presence of a variable topography. An analogous expression is obtained when deriving with respect to y.

Finally, the expression for the viscous term of the depth-averaged equation is obtained by integrating vertically the term $\nabla \cdot \boldsymbol{\tau}$ that appears in the Navier-Stokes Eq. Under the incompressible assumption, such term simplifies. We computed it in section §1.1.5.6 obtaining the expression $\mu\Delta\mathbf{u}$ in Eq. (1.50), where μ is the dynamic viscosity introduced in Eq. (1.26).

For the depth-averaged viscous term expression, consider the x-direction (it is similar for the y-direction case), then the integral results in

$$\int_B^{B+h} \mu \frac{\partial^2 u}{\partial z^2}\, dz = \mu \frac{\partial u}{\partial z}\bigg|_B^{B+h} = \mu \left[\frac{\partial u}{\partial z}(\mathbf{x}, B+h, t) - \frac{\partial u}{\partial z}(\mathbf{x}, B, t) \right] \overset{(*)}{=} -\mu\, \partial_z u(\mathbf{x}, B, t),$$

where in $(*)$ we have considered again no traction between air and fluid, as done for the parabolic velocity profile assumption. In particular, according with the parabolic profile expressed in Eq. (2.5), the vertical derivative evaluated at the bottom is

$$\partial_z u(\mathbf{x}, B, t) = 3\frac{U(\mathbf{x}, t)}{h}$$

so that the viscous term expression becomes

$$\int_B^{B+h} \mu \frac{\partial^2 u}{\partial z^2}\, dz = -\rho(\mathbf{x}, t)\gamma U(\mathbf{x}, t), \tag{2.11}$$

where $\gamma := \dfrac{3\nu}{h}$ is the friction coefficient, depending on kinematic viscosity and flow thickness (by reminding the relationship between dynamic and kinematic viscosity: $\mu = \rho\nu$). Note that Gerbeau and Perthame [101], by using a different approach, obtained a similar expression for γ. For a Bingham plastic rheology model (introduced in §1.1.5.7), the derivation of the viscous term is similar and it descends to be

$$\left(\frac{3}{h}\widetilde{\mu} + \frac{\tau_0}{|U|} \right) U,$$

where we used power-law viscosity $\widetilde{\mu}$ instead of the dynamic viscosity μ because the Bingham model uses the apparent viscosity (the reader may refer to Eqs. (1.51) and (1.52)). Hence, the factor γ in this case is defined as follows:

$$\gamma := \frac{3}{h\rho}\widetilde{\mu} + \frac{\tau_0}{\rho|U|}. \tag{2.12}$$

For several materials the dynamic viscosity μ is strongly temperature dependent, and in the case of lava a simple exponential relationship between magma viscosity and temperature can be assumed [53]:

$$\mu = \mu_{ref} \exp[-b(T - T_{ref})], \tag{2.13}$$

where b is an appropriate rheological parameter and μ_{ref} is the viscosity value at the reference temperature T_{ref} (for instance, $T_{ref} = T_{vent}$ with T_{vent} equal to the emission temperature at the vent). In the present work we do not explicitly account for crystallization and crystallinity-dependence of the viscosity, but they are implicitly considered in the determination of value of the rheological parameters b.

Momentum equation (complete form). To sum up, we write the complete expression of the *depth-averaged momentum equation* in the general two-dimensional setting, by using the results previously obtained: the transient term from Eq. (2.8), the advective flux expression from Eq. (2.9), the pressure contribution from Eq. (2.10) and the viscosity term from Eq. (2.11)

$$\frac{\partial(\rho h\mathbf{U})}{\partial t} + \nabla\cdot\left(\beta_u \rho h\mathbf{U}\mathbf{U}^T\right) + \nabla\left(\frac{1}{2}\rho g h^2\right) = -\rho g h\nabla B - \rho\gamma\mathbf{U}. \tag{2.14}$$

2.1.3 Thermal boundary layer

Variations in fluid temperature are of great importance when they affect the physical properties of the fluid and hence the dynamic itself. For example, this happens in the cooling process of lava because owing to the viscosity increase the lava slows down its motion; the same happens also for the wax because it has similar behavior. Even in the metalworking process, it is important knowing the cooling state of the molten material.

A hundred years ago, Prandtl [219] introduced the concept of boundary layer for a fluid flowing on a plate: the framework consists of a fluid moving horizontally, in laminar regime, initially with constant and uniform velocity and temperature, commonly denoted as u_∞ and T_∞, see Figure 2.3. When the fluid gets in touch with a horizontal and stationary plate uniformly cooled/heated to a different temperature, then the fluid starts to flow over the plate and the velocity and temperature profiles change. In the case of no-slip boundary condition, the fluid has zero velocity at the plate surface, but, moving further from it, the velocity increases gradually and asymptotically approaches the free stream velocity u_∞. Similarly, the temperature is cooler/hotter close to the plate, but, when approaching the free stream part of the fluid, it is near to the free stream temperature T_∞. These regions next to the plate, where velocity and temperature vary, are called *velocity* and *thermal boundary layers* respectively. The thermal boundary layer thickness is usually denoted as δ_T, and its common definition is the vertical distance from the plate to the point with a temperature that is the 99% of the free stream temperature. The thickness of such boundary layers changes in space, since it is thinner close to the edge of the stationary plate that is considered, and it increases moving from it, see Figure 2.3 and Schlichting [241].

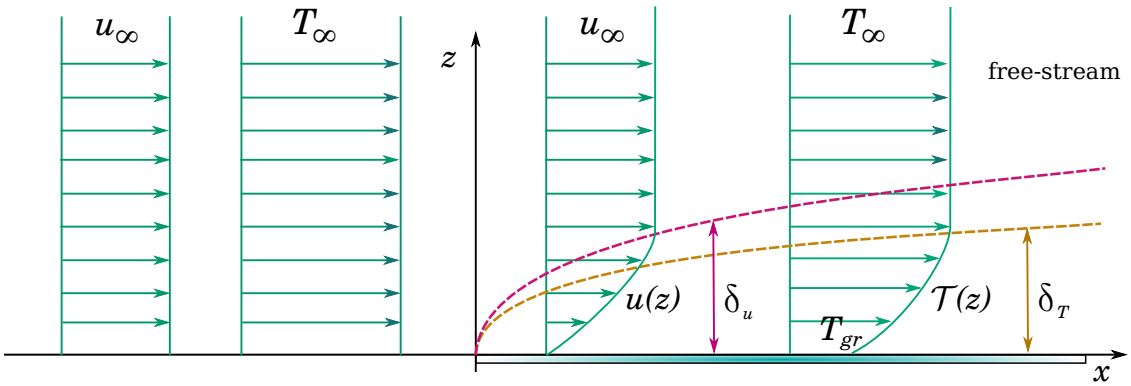


Figure 2.3: Velocity and thermal boundary layers on a cold flat plane. u_∞ and T_∞ denote the unchanging velocity and temperature of the free stream, T_{gr} indicates the temperature of the fluid in contact with the cold plate, $u(z)$ and $\mathcal{T}(z)$ are the vertical distributions of velocity and temperature, while δ_u and δ_T are the thicknesses of the velocity and thermal boundary layers.

In our model, we assume that the velocity boundary layer coincides with the whole fluid thickness, as represented in Figure 2.1, and that the thermal boundary layer is considered as a fixed fraction of the fluid depth. In the next sections §2.1.4 and §2.1.5 we present two different models that both account a linear temperature profile in the thermal boundary layer. The first one sees only advection and a fixed value for the temperature on the surface. Instead, in the second situation, we consider heat exchanges between the fluid and the environment, with the consequence that the surface temperature changes too.

2.1.4 Temperature profile with fixed surface value

Because of the dependence of friction and viscosity on temperature (as shown in §2.1.2), besides the Eqs. (2.7), (2.14) for the mass and momentum conservation, it is necessary to solve an additional equation describing the temperature evolution. The temperature equation is usually derived from the energy conservation law (written in terms of the temperature), but in our model we consider the temperature only as a transported quantity. For these motivations, we refer to a simple transport equation for $\mathcal{T}$:

$$\frac{\partial \mathcal{T}}{\partial t} + \nabla \cdot (\mathcal{T}\mathbf{u}) = 0. \tag{2.15}$$

We assume in this subsection two hypotheses: *(i)* the temperature admits a *thermal boundary layer* near the ground over a fixed fraction of the flow thickness, namely $\delta_T = h/n$ (with $n \geqslant 1$); *(ii)* a linear profile is adopted at the ground with a constant temperature for $z > B + \delta_T$, i.e. $\mathcal{T}(\mathbf{x}, B+h, t) = T_{surf}$, $\forall \mathbf{x}$, $\forall t$ (see Figure 2.4). One appends that this procedure is general and applicable to other variables for which similar hypotheses *(i)-(ii)* may hold, like for example to particle concentration in a sediment-laden flow, where a Rouse profile for the suspension might be used [235, 262].

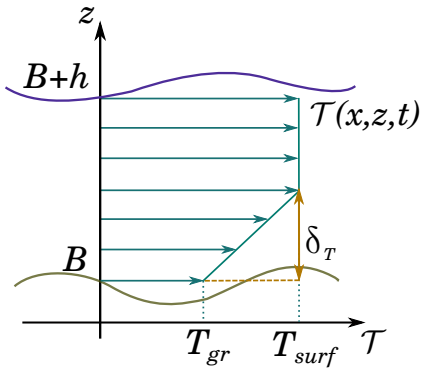


Figure 2.4: Piece-wise linear temperature profile. The thermal boundary layer δ_T is assumed linear, and a constant temperature T_{surf} at the surface is assumed.

The explicit expression of the assumed profile is

$$\mathcal{T}(\mathbf{x}, z, t) = \begin{cases} T_{surf}, & \text{if } B + \delta_T < z \leqslant B + h, \\ \dfrac{T_{surf} - T_{gr}(\mathbf{x}, t)}{\delta_T}(z - B) + T_{gr}(\mathbf{x}, t), & \text{if } B < z \leqslant B + \delta_T. \end{cases} \tag{2.16}$$

We compute the depth-averaged temperature T by using the definition of Eq. (2.2) and then we obtain a relation between T, T_{surf} and T_{gr}

$$\begin{aligned}
T &\overset{(2.2)}{=} \frac{1}{h}\int_B^{B+h} \mathcal{T}(z)dz \\
&= \frac{1}{h}\int_B^{B+\delta_T} \left[\frac{T_{surf}-T_{gr}}{\delta_T}(z-B)+T_{gr}\right]dz + \frac{1}{h}\int_{B+\delta_T}^{B+h} T_{surf}dz \\
&= \frac{1}{h}\left[\frac{T_{surf}-T_{gr}}{2\delta_T}(z-B)^2+T_{gr}z\right]_B^{B+\delta_T} + \frac{1}{h}\Big[T_{surf}z\Big]_{B+\delta_T}^{B+h} \\
&= \frac{1}{h}\left[\frac{T_{surf}-T_{gr}}{2\delta_T}\delta_T^2+T_{gr}\delta_T+T_{surf}(h-\delta_T)\right] \\
&= \frac{1}{h}\left[\frac{T_{surf}-T_{gr}}{2}\frac{h}{n}+T_{gr}\frac{h}{n}+T_{surf}\left(h-\frac{h}{n}\right)\right] \\
&= \frac{1}{2n}T_{gr}+\frac{2n-1}{2n}T_{surf},
\end{aligned}$$

and we use the result to express T_{gr} as a function of T and T_{surf}:

$$T_{gr}(\mathbf{x},t) = 2nT(\mathbf{x},t) + (1-2n)T_{surf}.$$

Thanks to the previous relation, the profile of $\mathcal{T}(\mathbf{x},z,t)$ (see Eq. (2.16)) may be rewritten in terms of the averaged temperature T, of the free-surface temperature T_{surf} and of the thickness index n as follows:

$$\mathcal{T}(\mathbf{x},z,t) = \begin{cases} \dfrac{2n\big[T_{surf}-T(\mathbf{x},t)\big]}{\delta_T}(z-B)+(1-2n)T_{surf}+2nT(\mathbf{x},t), & \text{if } B \leqslant z \leqslant B+\delta_T \\ T_{surf}, \quad \text{if } B+\delta_T < z \leqslant B+h. \end{cases} \tag{2.17}$$

By integrating over the depth the transient term of Eq. (2.15), we obtain

$$\int_B^{B+h} \frac{\partial \mathcal{T}(z)}{\partial t}dz = \frac{\partial}{\partial t}\int_B^{B+h}\mathcal{T}(z)dz = \frac{\partial(hT)}{\partial t}. \tag{2.18}$$

For the vertical integration of the advective term of Eq. (2.15), the use of the expression (2.17) for the piecewise linear temperature profile and of Eq. (2.5) for the parabolic velocity profile, lead to the sum of two terms whose coefficients depend on the parameter n related to the thermal boundary layer thickness h/n; the expression of the flux with respect to the x-direction is

$$\int_B^{B+h} \mathcal{T}(z)\,u(z)\,dz = \frac{4n^2-n}{4n^3}hTU + \frac{4n^3-4n^2+n}{4n^3}hT_{surf}U,$$

and the expression for the y-direction is similar. We denote by β_T the coefficient of the first term on the right-hand side of the previous equation and we obtain the 1D formulation of the advective flux

$$hU\big[\beta_T T + (1-\beta_T)T_{surf}\big]. \tag{2.19}$$

Being $0 \leqslant \beta_T \leqslant 1$, the term in brackets in the previous equation corresponds to a convex combination of the maximum and averaged temperatures. For instance, when

considering $n = 4$ (as suggested by Costa and Macedonio [55] for a lava model), the value of the coefficient is $\beta_T \approx 0.24$. One observes that the smaller is the thickness of the thermal boundary layer and the smaller is also the value of β_T. Thus, when the boundary layer thickness goes to zero (n goes to $+\infty$) the maximum temperature and the depth-averaged temperature coincide and the vertical integration of the advective term leads to

$$\nabla \cdot \big(hT\mathbf{U}\big), \tag{2.20}$$

which is the classical form resulting from the assumption of a uniform temperature profile (hence $T = T_{surf}$) and corresponds to the choice $\beta_T = 0$.

We mention also that if the parabolic profile for velocity is not assumed, or, otherwise, if we consider a constant temperature distribution over the depth, then the advective term derived would be Eq. (2.20).

2.1.5 Temperature profile for soil conduction and other heat exchanges

In this section, we derive a depth-averaged temperature equation which accounts both the transport of the temperature $\mathcal{T}$ and the heat exchange phenomena, such as conduction, convection and radiation. We assume that the vertical temperature variations are due to the conductive heat flux between the fluid and the ground with the consequent development of two thermal boundary layers in both materials. Indeed, in the case of lava flows or similar situations, the terrain (or the bottom) is colder than the fluid, so that a thermal boundary layer emerges in the fluid. At the same time, the ground temperature increases to a certain depth because of the presence of the hotter fluid, hence an underground thermal boundary layer develops too. In such context, we arrive to describe the entire temperature profile in terms of the fluid depth-averaged temperature and of the unchanged temperature underground, and the expression of the advective flux that we get is very similar to what we found previously in §2.1.4, because we are assuming similar shapes for the vertical profiles and one fixed value of temperature. Finally, we derive the radiative and convective terms, and the viscous heating term.

We make two hypotheses: (i) we assume that the fluid and the ground are both homogeneous and isotropic materials, and that their thermodynamic properties are temperature independent, (ii) we consider the temperature profile reached at the thermal equilibrium. The Fourier law (consult [81]) states that the conductive flux $\mathbf{q}_{cond}$, intended as the heat flux through a unit area per unit of time, is linearly proportional to the negative temperature gradient and to the thermal conductivity k

$$\mathbf{q}_{cond} = -k\nabla\mathcal{T}.$$

Since, in our model, the conductive heat flux occurs along the vertical direction, we have only the z-component of $\mathbf{q}_{cond}$:

$$q_{cond,z} = -k\frac{\partial\mathcal{T}}{\partial z}. \tag{2.21}$$

Because of the first assumption (i), the thermal conductivity of the fluid and of the ground, named as k_{fl} and k_{soil} respectively, are constant and do not depend on temperature. Thanks to the second assumption (ii), we refer to the heat equation at the stationary state condition, namely to the Laplace equation. As the temperature variations (due to

conduction) come about the vertical direction, then the Laplace equation reduces to one term equation, which solution is a linear function:

$$\frac{\partial^2 \mathcal{T}}{\partial z^2} = 0 \quad \Longrightarrow \quad \mathcal{T}(z) = Az + B.$$

This leads to a linear profile for temperature in each thermal boundary layer considered and therefore motivates the assumption already made in §2.1.4. According with this result, the two temperature profiles, in both thermal boundary layers at the fluid/bottom interface, are assumed linear, therefore the profile overall the fluid and terrain depths is piecewise-linear. We denote as δ_T and δ_{soil} the thermal boundary layer thicknesses of fluid and soil respectively, T_{surf} is the temperature of the free fluid surface, T_{gr} is the temperature at the fluid/ground interface, and T_{soil} is the unchanged temperature underground of the soil, see Figure 2.5. The profile explicit expression is:

$$\mathcal{T}(\mathbf{x}, z, t) = \begin{cases} T_{surf}(\mathbf{x}, t), & \text{if } B + \delta_T < z \leqslant B + h \\ \dfrac{T_{surf}(\mathbf{x}, t) - T_{gr}(\mathbf{x}, t)}{\delta_T}(z - B) + T_{gr}(\mathbf{x}, t), & \text{if } B < z \leqslant B + \delta_T \\ \dfrac{T_{gr}(\mathbf{x}, t) - T_{soil}}{\delta_{soil}}\big[z - (B - \delta_{soil})\big] + T_{soil}, & \text{if } B - \delta_{soil} < z \leqslant B \\ T_{soil}, & \text{if } z \leqslant B - \delta_{soil}. \end{cases} \tag{2.22}$$

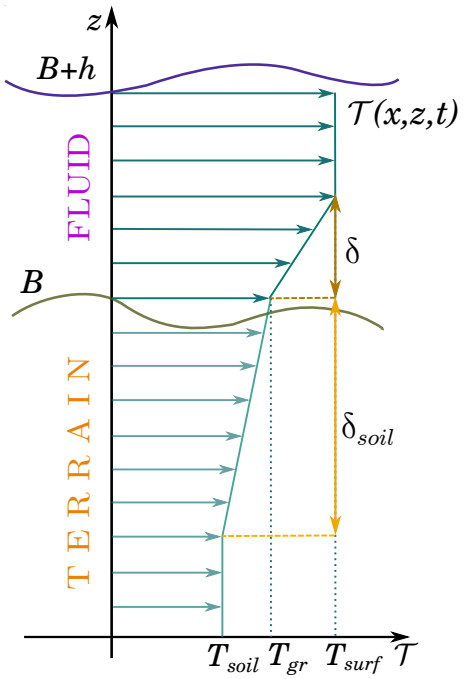


Figure 2.5: Vertical temperature profile. δ_T and δ_{soil} are the thermal boundary layers of fluid and ground respectively.

Usually, the thickness of the fluid thermal boundary layer is a fraction of the whole flow depth $\delta_T = h/n$ (with $n \geqslant 1$), meanwhile the thickness for the ground depends on the fluid depth as $\delta_{soil} = Mh$ (with $M \geqslant 1$); for example in the paper of Patrick et al. [209], which is a work about the characterization of the cooling of a stationary lava flow, the authors used $M = 2$.

We show in the following that the temperature at the fluid/ground interface T_{gr} depends on T_{soil}, on T_{surf}, on the physical properties of both materials and also on the thickness of the thermal boundary layers, according to the Fourier Law (2.21).

By assuming the same heat flux in both the means, the fluid and the solid, the following equality holds

$$q^{(fl)}_{cond,z} = -k_{fl}\frac{T_{surf}-T_{gr}}{\delta_T} = -k_{soil}\frac{T_{gr}-T_{soil}}{\delta_{soil}} = q^{(soil)}_{cond,z}. \tag{2.23}$$

From this equation we find the value of T_{gr} expressed in terms of the other parameters. First we rearrange the equation and we move on the left hand side T_{gr}

$$\left(\frac{k_{fl}}{\delta_T}+\frac{k_{soil}}{\delta_{soil}}\right)T_{gr} = \frac{k_{fl}}{\delta_T}T_{surf}+\frac{k_{soil}}{\delta_{soil}}T_{soil}$$
$$\left(\frac{k_{fl}}{\delta_T}\frac{\delta_{soil}}{k_{soil}}+1\right)T_{gr} = \frac{k_{fl}}{\delta_T}\frac{\delta_{soil}}{k_{soil}}T_{surf}+T_{soil}.$$

By renaming a coefficient as

$$\phi := \frac{1}{\frac{k_{fl}}{k_{soil}}\frac{\delta_{soil}}{\delta_T}+1} = \frac{1}{\frac{k_{fl}}{k_{soil}}nM+1}, \tag{2.24}$$

we find the expression for T_{gr} in terms of the other variables

$$T_{gr} = (1-\phi)T_{surf}+\phi T_{soil}. \tag{2.25}$$

From the definition of the depth-averaged temperature T, we find a relation binding T to the top and bottom fluid temperatures T_{surf} and T_{gr}, and to the thickness index n

$$T := \frac{1}{h}\int_B^{B+h}\mathcal{T}(z)dz = \frac{1}{2n}T_{gr}+\frac{2n-1}{2n}T_{surf} = aT_{gr}+(1-a)T_{surf}, \quad a := \frac{1}{2n}. \tag{2.26}$$

We want to express T_{surf} and T_{gr} in terms of the constant value T_{soil} and of the depth-averaged temperature T. First, we use the relations (2.25) and (2.26) to rewrite the expression of T

$$\begin{aligned} T &\overset{(2.26)}{=} aT_{gr}+(1-a)T_{surf} \\ &\overset{(2.25)}{=} a\big[(1-\phi)T_{surf}+\phi T_{soil}\big]+(1-a)T_{surf} \\ &= (1-a\phi)T_{surf}+a\phi T_{soil}, \end{aligned}$$

then we get the following expression for T_{surf} :

$$T_{surf} = \zeta T+(1-\zeta)T_{soil}, \quad \text{with } \zeta := \frac{1}{1-a\phi}. \tag{2.27}$$

In a similar way, we use the relations (2.25) and (2.27) to obtain T_{gr} as a function of T and T_{soil}

$$\begin{aligned} T_{gr} &\overset{(2.25)}{=} (1-\phi)T_{surf}+\phi T_{soil} \\ &\overset{(2.27)}{=} (1-\phi)\big[\zeta T+(1-\zeta)T_{soil}\big]+\phi T_{soil} \end{aligned}$$

so we find that

$$T_{gr} = \psi T+(1-\psi)T_{soil}, \quad \text{with } \psi := (1-\phi)\zeta = \frac{1-\phi}{1-a\phi}. \tag{2.28}$$

2.1.5.1 Advective term

We integrate with respect to the depth the advective flux of the temperature transport equation, Eq. (2.15). Without loss of generality we consider a flat bottom $B = 0$ and we recall from Eqs (2.5) and (2.22) the expressions of the parabolic velocity profile and of the piece-wise linear temperature profile for this simplified case.

$$\mathbf{u}(\mathbf{x}, z, t) = \frac{3}{2}\frac{\mathbf{U}}{h^2}\big(2hz - z^2\big),$$

$$\mathcal{T}(\mathbf{x}, z, t) = \begin{cases} T_{surf}(\mathbf{x}, t), & \text{if } \delta_T < z \leqslant h \\ \dfrac{T_{surf}(\mathbf{x}, t) - T_{gr}(\mathbf{x}, t)}{\delta_T} z + T_{gr}(\mathbf{x}, t), & \text{if } 0 < z \leqslant \delta_T. \end{cases}$$

We compute the depth-averaged advective flux along the x-direction, and similar steps are also valid for the y-direction

$$\begin{aligned}
\int_0^h u(z)\mathcal{T}(z)dz &= \int_0^{\delta_T} u(z)\mathcal{T}(z)dz + \int_{\delta_T}^h u(z)\mathcal{T}(z)dz \\
&= \int_0^{\delta_T} \left[\frac{3}{2}\frac{U}{h^2}\big(2hz - z^2\big)\left(\frac{T_{surf} - T_{gr}}{\delta_T} z + T_{gr}\right)\right] dz \\
&\quad + \int_{\delta_T}^h \left[\frac{3}{2}\frac{U}{h^2}\big(2hz - z^2\big) T_{surf}\right] dz \\
&= \frac{3}{2}\frac{U}{h^2}\left\{\left[\left(\frac{2}{3}hz^3 - \frac{1}{4}z^4\right)\frac{T_{surf} - T_{gr}}{\delta_T} + \left(hz^2 - \frac{1}{3}z^3\right) T_{gr}\right]_0^{\delta_T} \right. \\
&\quad \left. + \left[\left(hz^2 - \frac{1}{3}z^3\right) T_{surf}\right]_{\delta_T}^h \right\} \\
&= \frac{3}{2}\frac{U}{h^2}\left\{\left[\frac{2}{3}h\delta_T^2 - \frac{1}{4}\delta_T^3 - \left(h\delta_T^2 - \frac{1}{3}\delta_T^3\right)\right]\big(T_{surf} - T_{gr}\big) \right. \\
&\quad \left. + \frac{2}{3}h^3 T_{surf} \right\} \\
&= \frac{3}{2}\frac{U}{h^2}\left\{\left[\frac{1}{12}\delta_T^3 - \frac{1}{3}h\delta_T^2\right]\big(T_{surf} - T_{gr}\big) + \frac{2}{3}h^3 T_{surf} \right\} \\
&= \frac{3}{2}\frac{U}{h^2}\left\{\left[\frac{1}{12}\frac{h^3}{n^3} - \frac{1}{3}\frac{h^3}{n^2}\right]\big(T_{surf} - T_{gr}\big) + \frac{2}{3}h^3 T_{surf} \right\} \\
&= hU\left\{\left(1 - \frac{4n-1}{8n^3}\right) T_{surf} + \frac{4n-1}{8n^3} T_{gr} \right\}.
\end{aligned}$$

We rename the coefficient as θ, finding

$$\int_0^h u(z)\mathcal{T}(z)dz = hU\Big\{\big(1-\theta\big)T_{surf} + \theta T_{gr}\Big\}, \quad \theta := \frac{4n-1}{8n^3} \tag{2.29}$$

and then we write T_{surf} and T_{gr} in terms of T and T_{soil} by using (2.27) and (2.28)

$$\begin{aligned}
\int_0^h u(z)\mathcal{T}(z)dz &= hU\Big\{(1-\theta)\Big[\zeta T + (1-\zeta)T_{soil}\Big] + \theta\Big[\psi T + (1-\psi)T_{soil}\Big]\Big\} \\
&= hU\Big\{\Big[\theta\psi - \theta\zeta + \zeta\Big]T + \Big[1 - (\theta\psi - \theta\zeta + \zeta)\Big]T_{soil}\Big\}.
\end{aligned}$$

In the end, the depth-averaged advective flux term along the x-direction is:

$$hU\Big[\beta_T T + (1-\beta_T)T_{soil}\Big], \quad \beta_T := \theta\psi - \theta\zeta + \zeta. \tag{2.30}$$

In order to express the dependence of β_T on the physical quantities, we use the Eqs. (2.27), (2.28), (2.29) to replace the definitions of θ, ψ, ζ:

$$\begin{aligned}
\beta_T &= \theta\psi - \theta\zeta + \zeta \\
&= \frac{4n-1}{8n^3}\cdot\frac{1-\phi}{1-a\phi} - \frac{4n-1}{8n^3}\cdot\frac{1}{1-a\phi} + \frac{1}{1-a\phi} \\
&\overset{(2.26)}{=} \frac{4n-1}{8n^3}\cdot\frac{2n(1-\phi)}{2n-\phi} - \frac{4n-1}{8n^3}\cdot\frac{2n}{2n-\phi} + \frac{2n}{2n-\phi} \\
&= \frac{(4n-1)(1-\phi) - (4n-1) + 8n^3}{4n^2(2n-\phi)} \\
&= \frac{8n^3 - 4n^2\phi + 4n^2\phi - 4n\phi + \phi}{8n^3 - 4n^2\phi} \\
&= 1 + \frac{(2n-1)^2}{\frac{8n^3}{\phi} - 4n^2} \\
&\overset{(2.24)}{=} 1 + \frac{(2n-1)^2}{8n^3\left(\frac{k_{fl}}{k_{soil}}nM + 1\right) - 4n^2} \\
&= 1 + \frac{(2n-1)^2}{8n^4\left(\frac{k_{fl}}{k_{soil}}M + \frac{1}{8n} - \frac{1}{2n^6}\right)}.
\end{aligned}$$

From the last expression, we notice that the influence of the temperature of soil becomes negligible and the vertical distribution of temperature becomes similar to a constant vertical profile, in two situations: (i) when the thickness $\delta_T = h/n$ of the fluid thermal boundary layer thins, or (ii) when the thickness $\delta_{soil} = Mh$ of the soil thermal boundary layer increases a lot. In fact, as n or M goes to $+\infty$, the value of β_T gets closer to 1 so that the advective term reduces to $\nabla\cdot\big(hT\mathbf{U}\big)$ which is the classic expression obtained under the assumption of a uniform temperature profile (namely $T = T_{surf} = T_{gr}$). We notice also that when $n \to +\infty$ both ζ and ψ go to 1; in fact, for ζ we have

$$\begin{aligned}
\zeta &\overset{(2.27)}{=} \frac{1}{1-a\phi} \\
&\overset{(2.26)}{=} \frac{1}{1-\frac{\phi}{2n}} \\
&\overset{(2.24)}{=} \frac{1}{1-\frac{1}{2n^2\left(\frac{k_{fl}}{k_{soil}}M + \frac{1}{2n}\right)}} \underset{n\to+\infty}{\longrightarrow} 1,
\end{aligned} \tag{2.31}$$

and the same is immediately proved for ψ of Eq. (2.28) by following computations similar to what previously done for ζ. We get the consequence that $T_{surf} \approx T$ (from Eq. (2.27)) and $T_{gr} \approx T$ (from Eq. (2.28)).

The expression obtained in Eq. (2.30) is similar to that derived in the previous section, i.e. to Eq. (2.19), because we have assumed the same vertical profiles; moreover, the two temperatures that appear, in both cases, are the depth-averaged temperature $T(\mathbf{x}, t)$ and the unchanged temperature: in the former case there was the fixed surface temperature, in the latter case there is the fixed underground temperature. The observations done in §2.1.4 about the coefficient β_T and about the thickness of the thermal boundary layer are valid also in the present context.

2.1.5.2 Conductive heat transfer source term

In the previous section we analyzed the way that the conductive heat loss affects the temperature profile and we derived a consequent expression for the advective flux. We need to quantify the actual thermal loss due to conduction and to write the corresponding source term for the temperature equation. As already said by the Fourier Law (consult [81]), the heat exchanged between the hot fluid and cold ground, which affects only the boundary layer $\delta_T = h/n$, is directly proportional to the temperature differences and to the thermal conductivity, see Eq. (2.21) and Eq. (2.23). Since we are writing an equation for the temperature and not for the energy, we must divide the thermal conductivity by the specific heat c_p and by the density ρ ($\star$), obtaining that the thermal diffusivity κ is the coefficient appearing in the conductive source term:

$$q^{(fl)}_{cond,z} \overset{(2.23)}{=} -\frac{k_{fl}}{\delta_T}\Big[T_{surf} - T_{gr}\Big] \overset{(\star)}{\Longrightarrow} -\frac{k_{fl}}{\rho c_p \delta_T}\Big[T_{surf} - T_{gr}\Big] = -\frac{\kappa}{\delta_T}\Big[T_{surf} - T_{gr}\Big].$$

Reminding that the temperatures at the surface and at ground can be expressed in terms of T and T_{soil}, according with the relations (2.27), (2.28), we find that the previous formula becomes

$$-\frac{\kappa}{\delta_T}\Big[\zeta T + (1-\zeta)T_{soil} - \big(\psi T + (1-\psi)T_{soil}\big)\Big], \tag{2.32}$$

and, by rearranging the expression and renaming the conductive coefficients, we get the final term:

$$-\mathcal{H}\big(\zeta - \psi\big)\big(T - T_{soil}\big), \quad \mathcal{H} := \frac{\kappa n}{h}. \tag{2.33}$$

2.1.5.3 Convective heat transfer source term

The convection heat loss is the heat transfer from the surface of a warm body to the surrounding colder gas or liquid and is characterized by the motion of such fluid. Convection involves the combination of two processes, the heat diffusion and the advection of the warmed fluid. Convection might be a natural phenomenon driven by the buoyancy force, but also an artificial fact caused by a forced motion, or a mix of both of them. Natural convection, also said free convection, is driven by the buoyancy force and it is caused by the density variations due to the fluid warming. In natural convection, a hot body warms the fluids around it by heat diffusion, then the warmed fluid moves up because of the buoyancy forces and this induces the fluid advection, so the farther and colder fluid gets in touch with the hot body cooling it. This phenomenon is visible in everyday life when a pot with water is warmed on the stove. The fire heats the pot bottom which warms the water near there due to conductive heat transfer, the heat spreads in the fluid because of diffusion. As soon as the water near the bottom is warm enough, its density decreases (and the volume increases) hence it starts to move because of the buoyancy force, triggering the convective motion. In forced convection, the fluid advection is induced mechanically and

it produces the body cooling. In our case, the hot body corresponds to the hot free-surface fluid which is overhang by the colder air. In models where the air is not represented (as in our case), or when the whole complexity of the processes are not properly described, there is not the exact modeling of the convective phenomenon. However, the convective heat flux can be described by the Newton law of cooling in terms of the heat transfer coefficient λ and of the temperature difference between the fluid surface and the environment (see [81]) and writes as follows:

$$q_{conv} = \lambda f\big[T_{surf} - T_{env}\big].$$

The coefficient λ depends on the physical properties of the second fluid; for example, in the case of the lava that flows on the ground, the second fluid is the air, but we may also think about a submerged volcano, and in that circumstance the second fluid would be the water. Moreover, the value of λ changes if one models natural or forced convection, for example a value between 2.5–25 $\mathrm{Wm^{-2}\,K^{-1}}$ is proposed in literature [153] for natural air convection and between 10–500 $\mathrm{Wm^{-2}\,K^{-1}}$ for the forced case. The variable f indicates the fractional area of the exposed inner core: the value of f is exactly equal to 1 for a fluid completely molten; when there is a superficial crust, then it insulates partially or totally the fluid, as might happen in the case of lava (consult Fagents et al. [81] for more details). Moreover, in real flows, the value of f may change with time and space, because it depends on the solidification temperature of the material considered and on its chemical composition. Despite these things, in our model we assume for simplicity a constant value for f.

We need to divide the coefficient by the specific heat c_p and the density ρ of the fluid because we deal with a temperature equation; we rename the convective coefficient and obtain the source term to be added to the temperature equation:

$$-\mathcal{W}\big[\zeta T + (1-\zeta)T_{soil} - T_{env}\big], \quad \mathcal{W} := \frac{\lambda f}{\rho c_p}. \tag{2.34}$$

2.1.5.4 Thermal radiation transfer source term

The thermal radiation is the energy emitted by matter in form of electromagnetic waves and it involves every "hot body", namely every body which temperature is greater than the absolute zero temperature, that is $0\,\mathrm{K} = -273.15$ C; in fact, the movement of atoms and molecules of the body produces electromagnetic waves which transfer the energy from the surface of the body to the surrounding. The radiative heat flux is described by the Stefan-Boltzmann law, which name is due to two Austrian physicists, Josef Stefan and Ludwig Boltzmann, that formulated the same law as result of their different studies, in 1879 and 1884 respectively. The Stefan-Boltzmann law states that the rate of thermal radiation emitted from a surface per unit area q_{rad} is as follows

$$q_{rad} = \epsilon \sigma_{SB} T_{surf}^4, \tag{2.35}$$

namely it is proportional to the fourth power of its absolute temperature at the surface T_{surf} expressed in kelvin, to the Stefan-Boltzmann constant $\sigma_{SB} = 5.67 \cdot 10^{-8}$ and to the emissivity ϵ of the material, for more details see Modest [194]. The emissivity of the matter ϵ is a parameter that indicates how well the body radiates, for example, $\epsilon = 1$ for the "black body" (that is the ideal body that absorbs all incident electromagnetic radiation and that radiates back all the energy absorbed) and $\epsilon = 0$ for a body that is a perfect "reflector" (namely a body that does not emit nor absorb thermal radiation); for the real objects the value of ϵ falls between these two cases.

The energy balance for the "opaque surfaces" (namely the surfaces that are not perfect reflectors, i.e. with the emissivity $\epsilon \neq 0$) is the difference between the thermal radiation emitted and that absorbed. The absorption depends on the thermal radiations emitted from the surfaces around the observed body, including those far away from it. In our model of free-surface fluid, the thermal radiation of both the fluid and of the environment are accounted. As a consequence, the radiative heat flux depends on the difference of the fourth powers of the surface temperature and environmental temperature, according with Eq. (2.35), and it follows that the radiative heat loss of the fluid is

$$-\epsilon\sigma_{SB} f\Big[T_{surf}^4 - T_{env}^4\Big]$$

where T_{env} denotes the constant environmental temperature, ϵ is the emissivity of the fluid and the variable f indicates the fractional area of the exposed inner core, that appears because also the radiative heat loss is influenced by the presence of a superficial crust.

Since we deal with a temperature equation, as in the previous section we have to divide the heat loss by the specific heat c_p and the density ρ of the fluid to obtain the radiative term. We express again the surface temperature in terms of T and T_{soil}, according with Eq. (2.27), and rename the radiative coefficient, so we get the final expression for the radiative term in the transport equation:

$$-\mathcal{E}\Big[\big(\zeta T + (1-\zeta)T_{soil}\big)^4 - T_{env}^4\Big], \quad \mathcal{E} := \frac{\epsilon\sigma_{SB} f}{\rho c_p}. \tag{2.36}$$

Further considerations

We make some considerations about the temperature model derived so far.

- From comparing the types of heat transfer in the last three subsections, two main differences arise. The first one is that radiation does not require the presence of a medium to transfer the energy, whereas it is the opposite for conduction and convection. The second difference is about the temperature dependence: it is linearly proportional to the temperature difference for both conductive and convective heat transfers, whereas the radiative heat transfer is proportional to the difference of the fourth power of temperatures. As a consequence, the radiative transfer is more relevant when great temperature differences between the fluid and the ambient are present.

- In the case that a constant velocity profile is assumed, namely when $u(z) = U$, for example in the classic formulation of the shallow-water equations, a thermal boundary layer may still be adopted, with the consequence that the expression of the advective flux term simplifies into

$$\int_B^{B+h} u(z)\mathcal{T}(z)dz = \int_B^{B+h} U\mathcal{T}(z)dz = U\int_B^{B+h} \mathcal{T}(z)dz \overset{(2.2)}{=} UhT$$

and the terms rewarding the thermal exchange with the ground and the environment are Eqs. (2.33, 2.36, 2.34):

$$\frac{\partial(hT)}{\partial t} + \nabla\cdot(hT\mathbf{U}) = -\mathcal{H}\big(\zeta-\psi\big)\big(T - T_{soil}\big) - \mathcal{W}\big[\zeta T + (1-\zeta)T_{soil} - T_{env}\big] - \mathcal{E}\Big[\big(\zeta T + (1-\zeta)T_{soil}\big)^4 - T_{env}^4\Big].$$

- We consider the asymptotic case for $n \to \infty$. The convective term in Eq. (2.34) and the radiative term in Eq. (2.36) contain the parameter ζ that multiplies T and we have seen in Eq. (2.31) that ζ goes to 1 when n goes to ∞. Therefore, the asymptotic behavior of the convective and radiative terms is the following

$$-\mathcal{W}\Big[\zeta T + (1-\zeta)T_{soil} - T_{env}\Big] \underset{n\to+\infty}{\longrightarrow} -\mathcal{W}\left(T - T_{env}\right),$$
$$-\mathcal{E}\Big\{\big[\zeta T + (1-\zeta)T_{soil}\big]^4 - T_{env}^4\Big\} \underset{n\to+\infty}{\longrightarrow} -\mathcal{E}\left(T^4 - T_{env}^4\right).$$

  Concerning the conductive term, we consider the Eq. (2.32) where we recognize the expression of the ground temperature defined in Eq. (2.28). The parameter ψ too, that is present in the definition of T_{gr}, goes to 1 as n goes to $+\infty$, with the consequence that $T_{gr} \to T$. This produces that the term of the conductive heat loss goes to zero, which can be expected because we related the presence of heat conduction to the existence of the thermal boundary layer:

$$-\mathcal{H}\Big[\zeta T + (1-\zeta)T_{soil} - \underbrace{\big(\psi T + (1-\psi)T_{soil}\big)}_{T_{gr}}\Big] \underset{n\to+\infty}{\longrightarrow} -\mathcal{H}\left(T - T\right) = 0.$$

  If we aim to model the heat loss due to conduction even in the case of a constant temperature profile, we must define the temperature at the ground, which we would denote as $\widetilde{T}_{gr}$, in another way.

  Costa and Macedonio [55], for example, consider $\widetilde{T}_{gr}$ as a constant value, fixed a priori, that denotes the temperature of the fluid at the ground (which is smaller than the averaged temperature because of the conductive heat transfer with the soil). Also, the authors do not model the temperature profile explicitly (as we do), hence they only deal with the depth-averaged temperature T and write the source terms as follows:

$$-\mathcal{H}(T - \widetilde{T}_{gr}) - \mathcal{W}(T - T_{env}) - \mathcal{E}(T^4 - T_{env}^4) \tag{2.37}$$

  (actually, these terms coincide in a different order with those originally reported in the article [55]).

  We underline that, even though the assumption of constant temperature profile leads to a simpler model, the correct setting of the $\widetilde{T}_{gr}$ value is not obvious.

- In a simulation with a source area representing a vent, a shrewdness must be accounted. At the vent there is no cooling at the bottom because there is no conductive heat exchange with the ground. For this reason, only in the source area, the advective flux term assumes the classic expression and only the heat exchanges with the air must be considered, namely radiation and convection, hence only the second and third terms of equation (2.37)

$$\frac{\partial(hT)}{\partial t} + \nabla \cdot (hT\mathbf{U}) = -\mathcal{W}(T - T_{env}) - \mathcal{E}(T^4 - T_{env}^4).$$

2.1.5.5 Viscous heating term and effusive source term

In the dynamic of fluids characterized by a temperature-dependent viscosity, the coupling between the momentum and temperature (or energy) equations is important. In

our model we obtained such coupling by assuming the Nahme's exponential law of Eq. (2.13), in §2.1.2. In addition, we include also viscous heating, another important process that connects the temperature to the dynamic. For some viscous fluids, such as lava or polymers, the viscous friction produces an increase in temperature near the bottom, in the open-channel flow cases, or near the tube walls, when the fluid moves in a pipe, both natural and artificial. The temperature increment leads to a viscosity decrease, which reflects in a velocity increase, then the higher velocity causes further warming. The influence of the viscous heating on the dynamic is investigated in the work of Costa and Macedonio [54] where they highlight that this process is the cause of some unexpected phenomena. In the pahoehoe lava flows, for example, it produces that the temperature at the front is higher than the effusive one, as observed in Keszthelyi [147]. Moreover, the viscous heating may change the velocity and temperature profiles, for example, moving a parabolic velocity profile into a uniform velocity distribution, and this may happen both in the case of an open-channel flow (with the profile of Figure 2.1) and in the case of a flow in a natural or artificial conduit (with profile represented in Figure 2.2), see Kauahikaua et al. [141]. The viscous heating term is proportional to the dynamic viscosity and to the square of the vertical derivative of velocity. In our depth-averaged model, we compute it as follows

$$\begin{aligned}\frac{1}{\rho c_p}\int_B^{B+h}\mu\left[\left(\frac{\partial u}{\partial z}\right)^2+\left(\frac{\partial v}{\partial z}\right)^2\right]dz &\overset{(2.5)}{=}\frac{1}{\rho c_p}\int_B^{B+h}\mu\left[9\frac{\left(U^2+V^2\right)}{h^4}\left(z-(B+h)\right)^2\right]dz\\ &=\frac{3\mu}{\rho c_p h}\left(U^2+V^2\right)\end{aligned} \tag{2.38}$$

where we have used the analytic expression of the vertical derivative of $\mathbf{u}(z)$ defined in Eq. (2.5). When viscosity is temperature dependent, the viscous heating term writes as follows

$$\mathcal{K}\left(U^2+V^2\right)\exp\left[-b(T-T_{ref})\right],\quad \mathcal{K}:=\frac{3\mu_{ref}}{\rho c_p h} \tag{2.39}$$

where we have renamed the coefficient. If we had used a parabolic velocity profile for a velocity boundary layer of thickness $\delta_u = h/\alpha$, we would have found $\mathcal{K} = 3\alpha\mu/\rho c_p h$, instead we have simply $\alpha = 1$.

When one wants to consider also the release of new hot material in the system at the emissive temperature T_{vent}, an additional source term is required which accounts for the volumetric rate of fluid per unit area, R, namely the term

$$RT_{vent}.$$

2.1.6 Characteristic analysis

In summary, assuming an incompressible homogeneous fluid, a hydrostatic pressure distribution, a parabolic velocity profile, a piecewise linear temperature profile with a fixed temperature value on the surface (as derived in §2.1.4), the depth-averaged equations for

the flow in the absence of release of new material and with no heat exchange are:

$$\frac{\partial(\rho h)}{\partial t} + \nabla \cdot (\rho h \mathbf{U}) = 0, \tag{2.40a}$$

$$\frac{\partial(\rho h \mathbf{U})}{\partial t} + \nabla \cdot \left(\beta_u \rho h \mathbf{U}\mathbf{U}^T\right) + \nabla\left(\frac{1}{2}\rho g h^2\right) = -\rho g h \nabla B - \rho\gamma \mathbf{U}, \tag{2.40b}$$

$$\frac{\partial(hT)}{\partial t} + \nabla \cdot \left[\left(\beta_T T + (1-\beta_T)\, T_{surf}\right) h\mathbf{U}\right] = 0, \tag{2.40c}$$

where we underline that $\mathbf{U}$ is a column vector; the system falls into the framework presented in §1.1.6 and is coupled with the state equation that considers the density dependence on the average temperature

$$\rho(T) = mT + \rho_0. \tag{2.41}$$

Moreover, if we consider the heat exchanges between fluid, ground and environment, the viscous heating and the release of new fluid, then the temperature equation (2.40c) must be replaced by the results derived in §2.1.5 and the source term ρR must be added on the right hand side of the continuity equation

$$\frac{\partial(\rho h)}{\partial t} + \nabla \cdot (\rho h \mathbf{U}) = \rho R, \tag{2.42a}$$

$$\frac{\partial(\rho h \mathbf{U})}{\partial t} + \nabla \cdot \left(\beta_u \rho h \mathbf{U}\mathbf{U}^T\right) + \nabla\left(\frac{1}{2}\rho g h^2\right) = -\rho g h \nabla B - \rho\gamma \mathbf{U}, \tag{2.42b}$$

$$\begin{aligned}\frac{\partial(hT)}{\partial t} + \nabla \cdot \left[\left(\beta_T T + (1-\beta_T)\, T_{soil}\right) h\mathbf{U}\right] &= -\mathcal{H}\left(\zeta - \psi\right)\left(T - T_{soil}\right) \\ -\mathcal{W}\left[\zeta T + (1-\zeta)T_{soil} - T_{env}\right] &- \mathcal{E}\left[\left(\zeta T + (1-\zeta)T_{soil}\right)^4 - T_{env}^4\right] \\ &+\mathcal{K}\left(U^2 + V^2\right)\exp\left[-b(T - T_{ref})\right] + RT_{vent}.\end{aligned} \tag{2.42c}$$

Notice that no "vent" term is included in the momentum equations because we assume that the lava is emitted with no velocity along x and y.

The stationary steady-state solutions of our system are the *lake-at-rest* conditions:

$$\mathbf{U} = 0, \quad B + h = \text{const}, \quad T = \text{const}, \quad \rho = \text{const}, \tag{2.43}$$

in the case that there is no emission of new fluid in the system and when there are no heat exchanges with the environment.

The homogeneous part of the systems (2.40) or (2.42) are constituted by non-linear hyperbolic PDEs, and thus they are solved by classical numerical techniques developed for such kind of equations [168, 258]. In order to do that, we first rewrite the homogeneous system in a more compact notation, introducing the vector of *conservative variables* $\mathbf{q}$

$$\mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} := \begin{bmatrix} \rho h \\ \rho h U \\ \rho h V \\ hT \end{bmatrix}.$$

If we denote by $\mathbf{f} = (\mathbf{G}, \mathbf{H})$ the flux vector (argument of the divergences), and we express the fluxes as function of the conservative variables

$$\mathbf{G}(\mathbf{q}) = \begin{bmatrix} \rho h U \\ \beta_u \rho h U^2 + \dfrac{\rho g}{2} h^2 \\ \beta_u \rho h U V \\ \left(\beta_T T + (1-\beta_T) T_*\right) h U \end{bmatrix}, \quad \mathbf{H}(\mathbf{q}) = \begin{bmatrix} \rho h V \\ \beta_u \rho h U V \\ \beta_u \rho h V^2 + \dfrac{\rho g}{2} h^2 \\ \left(\beta_T T + (1-\beta_T) T_*\right) h V \end{bmatrix},$$

(where T_* represents T_{surf} or T_{soil}) then we rewrite the homogeneous part of the governing equations as

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot (\mathbf{f}(\mathbf{q})) \equiv \frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{G}(\mathbf{q})}{\partial x} + \frac{\partial \mathbf{H}(\mathbf{q})}{\partial y}. \tag{2.44}$$

In order to study the hyperbolicity and local propagation velocities, we replace the homogeneous system by the equivalent quasi-linear formulation

$$\mathbf{q}_t + \nabla \cdot (\mathbf{f}(\mathbf{q})) = 0 \quad \longleftrightarrow \quad \mathbf{q}_t + \mathbf{G}'(\mathbf{q})\mathbf{q}_x + \mathbf{H}'(\mathbf{q})\mathbf{q}_y = 0,$$

where $\mathbf{G}'$ and $\mathbf{H}'$ are the Jacobian matrices of the advective fluxes, and they read as follows:

$$\mathbf{G}'(\mathbf{q}) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\beta U^2 + gh & 2\beta U & 0 & 0 \\ \beta UV & \beta V & \beta U & 0 \\ -\beta_T(UT)/\rho & \beta_T T/\rho + (1-\beta_T)T_*/\rho & 0 & \beta_T U \end{bmatrix},$$

$$\mathbf{H}'(\mathbf{q}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ \beta UV & \beta V & \beta U & 0 \\ -\beta V^2 + gh & 0 & 2\beta V & 0 \\ -\beta_T(VT)/\rho & 0 & \beta_T T/\rho + (1-\beta_T)T_*/\rho & \beta_T V \end{bmatrix}.$$

The two Jacobians has four eigenvalues, which are the elements of these spectra:

$$\begin{aligned} Sp\,(\mathbf{G}') &= \left\{ \beta_u\, U \pm \sqrt{\beta_u(\beta_u - 1)U^2 + gh},\ \beta_u\, U,\ \beta_T U \right\}, \\ Sp\,(\mathbf{H}') &= \left\{ \beta_u\, V \pm \sqrt{\beta_u(\beta_u - 1)V^2 + gh},\ \beta_u\, V,\ \beta_T V \right\}. \end{aligned} \tag{2.45}$$

When the eigenvalues are real, it ensues that the equations are hyperbolic, and the perturbations in the solution propagate with finite velocities given by the previous equations. We also note that, when $\beta_u = 1$ (resulting from the assumption of a uniform velocity profile), the first eigenvalues reduce to

$$U \pm \sqrt{gh} \quad \text{and} \quad V \pm \sqrt{gh}, \tag{2.46}$$

thus they coincide with the characteristic velocities of the classical shallow water equations.

Recall that the *Froude number* is a dimensionless variable accounting for the relative importance of inertial and gravitational forces. Classically, the Froude number for shallow water waves is defined as

$$Fr = \frac{|\mathbf{U}|}{\sqrt{gh}}, \tag{2.47}$$

see LeVeque [168], and according to the value assumed by this ratio, the regime is classified as subcritical if $Fr < 1$, critical if $Fr = 1$ and supercritical if $Fr > 1$. When flow conditions are supercritical, surface waves generated by downstream disturbances cannot travel upstream. Conversely, when flow is subcritical, disturbance propagates both upstream and downstream. This classification is relevant also for numerical simulations, because the boundary conditions must be prescribed according to the flow regime. We

point out that the effects of a parabolic velocity profile assumption on the solution become less important as the Froude number approaches 0, because in such situations the dynamics are less affected by the inertial term.

As shown above, the parabolic velocity profile defined in Eq. (2.5) leads to the presence of the coefficient β_u in the inertial term appearing in the momentum equation. Because of that, from a mathematical point of view, also the critical conditions change with respect to the classical shallow water equations. In fact, considering the 1D case, the eigenvalues of the flux Jacobian for the parabolic velocity case have the same sign when the following condition is satisfied:

$$\frac{|U|}{\sqrt{(1-1/\beta_u)U^2 + gh/\beta_u^2}} > 1.$$

According to this formulation, for the same value of the flow thickness h, the critical regime for a parabolic profile is obtained at a smaller depth-averaged flow velocity than for constant velocity profile. In addition, we observe that the left-hand side quantity converges to the Froude number when the velocity tends to have constant profile, namely when β_u goes to 1.

2.1.7 The case of a density profile (outline)

What would happen in the case that a non-constant density profile was assumed? Without providing complete nor exhaustive discussions on this theme, we give two cues that might lead to future work developments.

In the assumptions of our model, we supposed that density has a linear dependence from temperature $\rho(z) = \rho_0 + m\mathcal{T}(z)$ with $m < 0$ (see Eq. (2.3)), and we narrowed it down to the cases where the vertical variations of density are negligible and hence we used the depth-averaged value $\bar{\rho}$ to derive the equations. What would happen to the equations if such restriction was deleted? We started to investigate in that direction and we present the preliminary results which accounts only the hyperbolic terms of mass and momentum equations. The piece-wise linear profile of temperature reflects into a piece-wise linear profile for density, as represented in Figure 2.6.

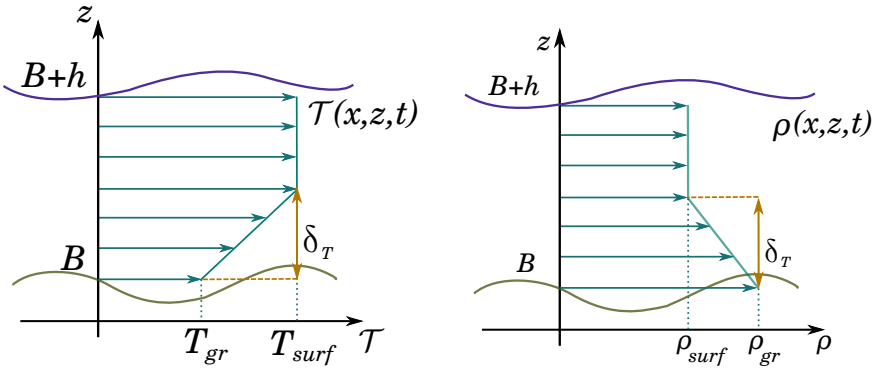


Figure 2.6: Vertical temperature and density profiles. *Left:* temperature piece-wise linear profile, *right:* density piece-wise linear profile that linearly depends on temperature.

If we consider the temperature model of §2.1.4, we may assume that the expression of the

density profile is

$$\rho(\mathbf{x}, z, t) = \begin{cases} \rho_{surf}, & \text{if } B + \delta_T < z \leqslant B + h, \\ \dfrac{\rho_{surf} - \rho_{gr}(\mathbf{x}, t)}{\delta_T}(z - B) + \rho_{gr}(\mathbf{x}, t), & \text{if } B < z \leqslant B + \delta_T, \end{cases} \tag{2.48}$$

where δ_T is exactly the thickness of the thermal boundary layer, and ρ_{surf} and ρ_{gr} denote the density at the surface and at the ground respectively and they implicitly depend on temperature. Similarly to the procedure adopted in §2.1.4, we can express the density profile in function of $\bar{\rho}$ and ρ_{surf}. Without entering in details, the transient and the advective terms of the mass and momentum equations have the following structures (for simplicity we present the 1D formulation)

$$\frac{\partial(\bar{\rho}h)}{\partial t} + \frac{\partial}{\partial x}\left[(\alpha_1\rho_{surf} + \alpha_2\bar{\rho})\, hU\right],$$
$$\frac{\partial}{\partial t}\left[(\alpha_1\rho_{surf} + \alpha_2\bar{\rho})\, hU\right] + \frac{\partial}{\partial x}\left[(\beta_1\rho_{surf} + \beta_2\bar{\rho})\, hU^2\right],$$

assuming also the parabolic profile for velocity, otherwise we would obtain the classic expression of the shallow water equation.

The next step of our work in this direction will be the derivation of the pressure term. Looking in literature, we found that the recent work of Pokrajac et al. [217] pursues a similar aim. The authors focus mainly on the pressure term derivation in the case of different density profiles, i.e. constant, linear and exponential, that are represented in Figure 2.7, but they do not get the explicit expressions of the other terms, neither they dig into the cases of co-presence of different profiles for the other variables.

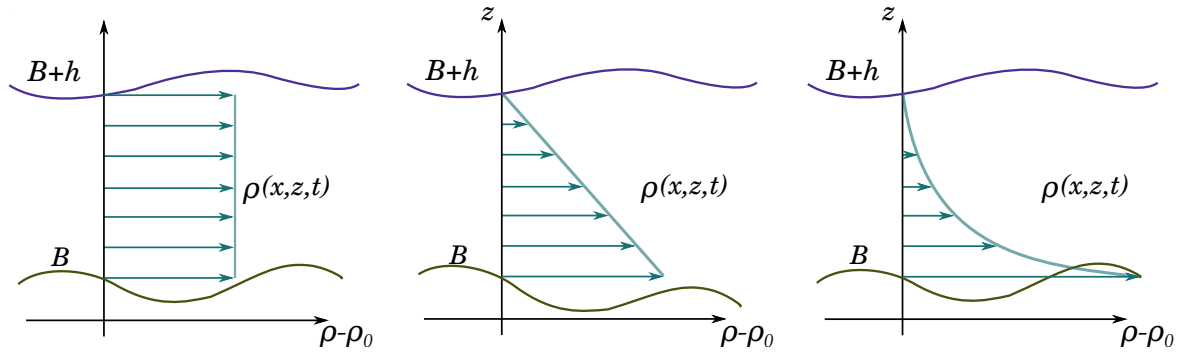


Figure 2.7: Vertical density profiles assumed in Pokrajac et al. [217] where ρ_0 is the ambient density. *Left:* constant, *center:* linear, *right:* exponential.

The argument carried out in Pokrajac et al. [217] leads to a coefficient in the pressure term that depends on the density profile or, equivalently, on the pressure profile. For an easier derivation of their model, they take into account also the ambient fluid, which has constant density ρ_0 and a pressure distribution $p_0(z)$, and they assume that on the free-surface of the fluid the pressure coincides with the ambient pressure. They consider a hydrostatic pressure distribution (as we do in §2.1.2):

$$p(z) - p_0(z) = \int_z^{B+h} \big(\rho(\zeta) - \rho_0\big) g d\zeta.$$

The pressure term that they obtain is

$$\frac{\partial}{\partial x}\left[\frac{1}{2}a_p\left(\bar{\rho}-\rho_0\right)gh^2\right]+\left(\bar{\rho}-\rho_0\right)gh\frac{\partial B}{\partial x} \tag{2.49}$$

which differs from ours (see Eq. (2.10)) in the presence of the coefficient a_p. Such coefficient is defined as the pressure force per unit width, normalized with the force that corresponds to the constant density:

$$a_p=\frac{\int_B^{B+h}\left(p(z)-p_0(z)\right)dz}{\frac{1}{2}\left[\frac{1}{h}\int_B^{B+h}\left(\rho(z)-\rho_0\right)dz\right]gh^2}=\frac{\int_B^{B+h}\left(p(z)-p_0(z)\right)dz}{\frac{1}{2}(\bar{\rho}-\rho_0)gh^2}=2\int_0^1\frac{p(\hat{z})-p_0(\hat{z})}{(\bar{\rho}-\rho_0)gh}d\hat{z} \tag{2.50}$$

where the last equality comes from the variable change $\hat{z}:=(z-B)/h$. In order to understand how the value of a_p varies according to the density profile, we analyze Figure 2.8 that depicts the integrated function with respect to $\hat{z}$, for the different density distributions. The value of a_p depends on the area under the functions represented, therefore it is easy to observe that a_p is equal to 1 for a constant density profile and it is less than 1 in the other cases. From this, it is evident that the pressure force (see Eq. (2.49)) is smaller for the variable density cases with respect to the constant density case. As a consequence, if the pressure coefficient a_p is omitted in the variable density cases, then the pressure force is overestimated.

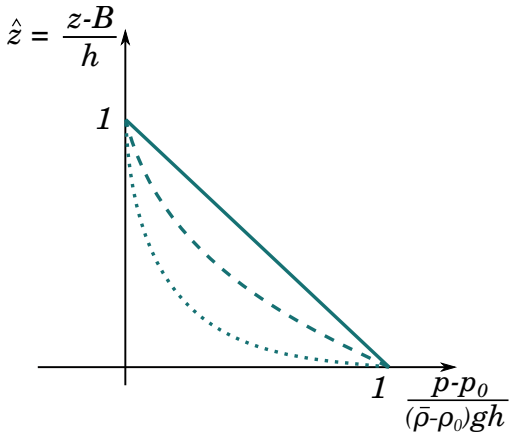


Figure 2.8: Vertical profile of the normalized pressure (the integrated function in Eq. (2.50), on the abscissa) with respect to the normalized depth: the *continuous line* corresponds to constant density profile, the *dashed line* to the linear density profile, the *dotted line* to the exponential density profile assumed in Pokrajac et al. [217].

The authors conclude that with the hypothesis of constant density there is an overestimation of the driving force due to the pressure gradient of 33% with respect to a linear profile and up to the 50% with respect to the exponential profile. There are consequences also for the propagation speed of the small disturbances because the coefficient a_p appears also in the eigenvalues of the Jacobians (see Eq. (2.46)), it follows that there is an overestimation of the local speeds of propagation of the 22% and 40% compared to linear and exponential profiles respectively.

Concerning our model and applications, we should follow what they did, namely determine the coefficient a_p, but using our density profile.

2.2 3D multiphase model

In this section, we present the equations that describe a 3D model. The advantage of having a 3D model is that the vertical profiles of variables are not forced a priori with any analytic assumption, but are free to change only according to the boundary conditions, the evolution of the dynamics, and the properties of the fluid. Our specific interest is in liquids with a free surface, and we choose a way to model them which takes into account also the air above. This means that we deal with a *multiphase* model for the liquid-gas couple. Moreover, the two fluids are considered immiscible. The multiphase modeling is realized by adopting the *Volume of Fluid* (VOF) method [127], where a new transport equation is added to the system of the governing equations, involving a new variable that describes the transport of the volume fraction of one phase. We treat both the liquid and the air above it as incompressible fluids, and hence they may be described by the same kind of equations, namely the incompressible Navier-Stokes Eqs. (1.49) (derived in section §1.1.5.6) that we report also here:

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0, \\ \frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot \left(\rho \mathbf{u}\mathbf{u}^T\right) &= \rho \mathbf{g} - \nabla p + \nabla \cdot \boldsymbol{\tau}, \end{aligned}$$

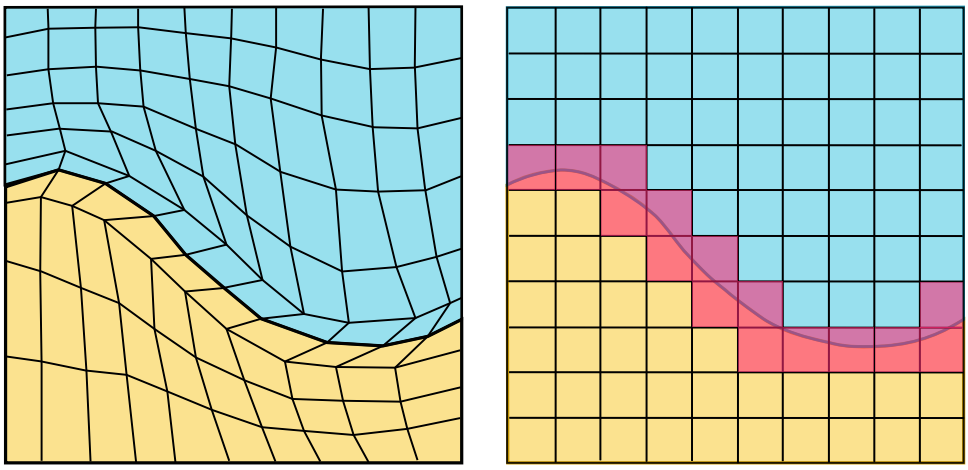
where we recall that ρ is the density, $\mathbf{u}$ is the velocity, $\mathbf{g}$ is the gravity acceleration, p is the pressure, and $\boldsymbol{\tau}$ is the viscous stress tensor defined in Eq. (1.47). In our model, we also assume that both liquid and gas have constant density respectively, therefore the continuity equation for the conservation of the mass reduces to the kinetic condition of null velocity divergence $\nabla \cdot \mathbf{u} = 0$ (as already observed in Eq. (1.41)). Even though we consider constant densities for both fluids, those are not equal, therefore we retain the expression of the momentum equation as that one reported above (instead of using the expression of Eq. (1.48) which applies in the constant density case). The new transport equation for the volumetric fraction of one phase allows us to distinguish where the two fluids are located, and to associate their densities correctly. In this 3D model that describes two immiscible fluids, we must consider also the effect of the surface tension; therefore an additional source term is added in the momentum equation. Being the lava flows the ultimate application of our model, we also have to enrich our system by an additional equation for the transport of the thermal energy, derived as a modified version of the energy conservation equation (1.37c).

Even though our multiphase (liquid-gas) 3D model describes two incompressible, immiscible, constant density fluids that exchange heat, our attention is mostly on the good thermodynamic description of the liquid phase. The section is organized in three parts. We start in §2.2.1 by giving an overview of the existing techniques that deal with the modeling of two immiscible multiphase fluids, and in §2.2.1.1 we focus on the derivation of the equation that characterizes the VOF method, which is the specific technique we decided to use. In §2.2.2 we show how to derive the transport equation for the thermal energy as a modified version of Eq. (1.37c) for the conservation of energy in the Navier-Stokes system, accounting for the radiative and convective heat transfer. The last section §2.2.3 exhibits the whole system of equations of our 3D model.

2.2.1 Immiscible multiphase simulations

In the description of the motion of two *immiscible* fluids, a good treatment of the free surface that separates them is of overall importance. Generally, one refers to two fluids belonging to different phases as liquid and gas, where the water-air couple is probably the most frequent case to simulate, but one may also consider the case of two immiscible liquids like water-oil or water-mercury. The free surface that separates the immiscible fluids is often called *phase interface* or simply *interface* and its evolution is computed as a part of the solution. Having in mind that the equations of any model we choose will be solved on a discrete computational grid, it is important to classify in two major groups the methods that treat the phase interface, since this leads to different modeling issues.

- The **interface-tracking methods** describe the free surface as a sharp boundary between the phases through the computational grid [126, 171, 223, 257]. In this family of methods the computational grid evolves in time and adapts and advances forward to define the free surface as represented in Figure 2.9.a; in particular, in each control volume only one phase can be present. This issue impacts on the model, where appropriate boundary conditions between the phases are applied directly to the surfaces of the domain belonging and defining the interface;
- The **interface-capturing methods** define the interface as those cells filled by both phases as shown in Figure 2.9.b, and the computational grid is possibly fixed, at most it is refined; therefore the interface is not as sharp as in the previous family of methods. As we will see, this leads to adding a further equation to the mathematical model.



(a) *Interface-tracking method.* (b) *Interface-capturing method.*

Figure 2.9: Computational grid and phase interface between two fluids.

In this work we focus on one specific method, proposed by Hirt and Nicholls [127] and called the **Volume of Fluid** (VOF) method; it belongs to the second group of the *interface-capturing methods*. See in §2.3 a brief overview of other methods in such family with the related modeling issues. A new variable α is introduced in VOF to describe the volume fraction of one phase and a dedicated transport equation is derived from the mass conservation equation

$$\partial_t \alpha + \nabla \cdot (\alpha \mathbf{u}) = 0.$$

When a cell is completely full with one of the two fluids then α is equal to one or zero respectively and the interface is defined by those cells that see both phases inside and hence a value $0 < \alpha < 1$, see Figure 2.10.

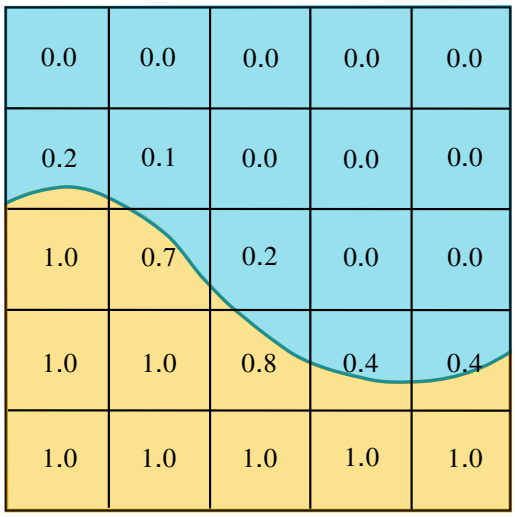


Figure 2.10: *Volume of Fluid.* The value assumed by the variable α is reported in each cell.

The Volume of Fluid approach can be applied with three different declinations. (i) The first one is that introduced by Hirt and Nicholls [127] and it sees the transport equation for α solved on the entire (multiphase) domain, instead the conservation equations for mass and momentum are solved only for the liquid phase, hence the gas phase behaves in an unrealistic way. (ii) Kawamura and Miyata [143] proposed the second variant in which a different transport equation for the density function is solved (instead of the volumetric fraction α they consider its product with the density $\rho\alpha$) and the free surface is the iso-surface with $\alpha = 0.5$. The conservation equations for mass and momentum are solved separately for the two phases and the interface is considered a boundary for both of them and boundary conditions are prescribed on it. This method was used particularly to compute the flows around ships and submerged bodies. (iii) For the third and last variant of the Volume of Fluid method, the two fluids are treated as a single fluid which properties (namely, density and viscosity) vary in space according to the volumetric fraction of each phase

$$\rho = \alpha\rho_l + (1-\alpha)\rho_g, \quad \mu = \alpha\mu_l + (1-\alpha)\mu_g,$$

and both fluids share the same velocity field and pressure field, hence the conservative equations for mass and momentum are solved for such single and promiscuous fluid. Since we follow this last variant of the VOF method, its whole derivation is shown in the next section according to Deshpande et al. [69].

2.2.1.1 Derivation of α-equation

Assume that a domain Ω sees the co-presence of two immiscible fluids, which from now on are referred to, for simplicity, as liquid and gas, and that those occupy two disjoint regions of the domain $\mathcal{R}_l(t)$ and $\mathcal{R}_g(t)$ respectively (varying with time), where

$$\mathcal{R}_l(t) \cup \mathcal{R}_g(t) = \Omega \quad \text{and} \quad \mathcal{R}_l(t) \cap \mathcal{R}_g(t) = \varnothing \ \forall t,$$

and the indicator-function of the liquid phase is defined accordingly

$$\chi_l(\mathbf{x},t) = \begin{cases} 1 & \text{if} \quad \mathbf{x} \in \mathcal{R}_l(t), \\ 0 & \text{if} \quad \mathbf{x} \in \mathcal{R}_g(t). \end{cases}$$

The densities of the two fluids are ρ_l and ρ_g (supposed to be constant in time) and, thanks to the immiscibility hypothesis, a shared density function is defined, that leans on the indicator-function of the liquid phase:

$$\begin{aligned} \rho(\mathbf{x},t) &= \rho_l \chi_l(\mathbf{x},t) + \rho_g\big(1-\chi_l(\mathbf{x},t)\big) \\ &= (\rho_l - \rho_g)\chi_l(\mathbf{x},t) + \rho_g. \end{aligned} \tag{2.52}$$

One may interpret that as if only one incompressible fluid is considered, in the whole domain, which density is distributed according to equation (2.52).

When a control volume $V \subset \Omega$ is considered, the function α of the liquid-phase volumetric fraction is introduced, defining it as the average over the whole volume of the indicator-function of the liquid phase, $\forall \mathbf{x} \in V$, $\forall V \subset \Omega$, as follows:

$$\alpha(\mathbf{x},t) = \frac{1}{|V|}\int_V \chi_l(\cdot,t)dV, \tag{2.53}$$

and it results as a piece-wise constant function. If a control volume is fully occupied by the liquid or by the gas, then the values assumed by α are exactly equal to one or to zero respectively, otherwise $0 < \alpha < 1$. Similarly, one defines another density function that is a piece-wise constant function named $\tilde{\rho}$ which corresponds to the density averaged over each control volume, it is defined $\forall \mathbf{x} \in V$, $\forall V \subset \Omega$, and can be expressed in terms of the liquid volumetric fraction α as follows:

$$\begin{aligned} \tilde{\rho}(\mathbf{x},t) &= \frac{1}{|V|}\int_V \rho(\cdot,t)dV \\ &\overset{(2.52)}{=} \frac{1}{|V|}\int_V (\rho_l-\rho_g)\chi_l(\cdot,t)dV + \frac{1}{|V|}\int_V \rho_g dV \\ &\overset{(2.53)}{=} (\rho_l-\rho_g)\alpha(\mathbf{x},t) + \rho_g \\ &= \rho_l\alpha(\mathbf{x},t) + \rho_g(1-\alpha(\mathbf{x},t)). \end{aligned} \tag{2.54}$$

From the conservation of the mass (whose equation is averaged in every control volume V) descends the transport equation for the average density

$$\partial_t\tilde{\rho} + \nabla\cdot(\tilde{\rho}\mathbf{u}) = 0 \tag{2.55}$$

where $\mathbf{u}$ is a velocity function shared by both phases through the entire domain. From the combination of the density expression (2.54) with the mass conservation equation (2.55) descends that

$$\partial_t\big((\rho_l-\rho_g)\alpha(\mathbf{x},t)\big) + \partial_t\rho_g + \nabla\cdot\big((\rho_l-\rho_g)\alpha(\mathbf{x},t)\mathbf{u}(\mathbf{x},t)\big) + \nabla\cdot\big(\rho_g\mathbf{u}(\mathbf{x},t)\big) = 0$$

and by reminding that the flow field is divergence free ($\nabla\cdot\mathbf{u} = 0$) due to the incompressibility constraint and that ρ_g is constant in time, we obtain a transport equation for the liquid volume fraction

$$\partial_t\alpha(\mathbf{x},t) + \nabla\cdot\big(\alpha(\mathbf{x},t)\mathbf{u}(\mathbf{x},t)\big) = 0 \tag{2.56}$$

that is the principal equation in the VOF method.

The presence of two fluids entails that even the **surface tension** is considered. It is modeled as a continuum surface force [27] and added in the momentum equation defining it as:

$$\mathbf{f}_\Sigma := \sigma_\Sigma \kappa_\Sigma \nabla\alpha, \quad \kappa_\Sigma = -\nabla \cdot \frac{\nabla\alpha}{|\nabla\alpha|}, \tag{2.57}$$

where σ_Σ is the surface tension constant and κ_Σ is the curvature of the surface.

2.2.2 Energy equation

In our 3D model, we consider a modified version of the equation for the energy conservation, whose general form is written in Eq. (1.37c). Such modified equation goes to join the system constituted by the mass and momentum conservation equations and by the liquid volume fraction equation. We consider only the thermal energy, namely $E = \rho c_p T$, where T is the temperature and c_p is the specific heat of the fluid. The hyperbolic terms on the left hand side of the energy conservation Eq. (1.37c) are retained, i.e. $\partial_t E + \nabla \cdot (E\mathbf{u})$, together with the conductive flux term $\nabla \cdot \mathbf{q}_{cond}$. We assume that (i) the pressure variations produce negligible effects on the thermodynamic variables and that (ii) the effects of the energy dissipation caused by viscosity are small enough to be neglected. Because of these two hypotheses, that are common to many contexts including lava flows, the pressure and the viscous terms that were present in Eq. (1.37c) are not accounted. From the Fourier Law of heat conduction (see §1.1.5.4, §2.1.5 and [81] for more details), the conductive flux is $\mathbf{q}_{cond} = -k\nabla T$, then it descends that the term involving the divergence of the conductive flux, that appears in the equation, becomes a Laplacian term as follows

$$-\nabla \cdot \mathbf{q}_{cond} = -\nabla \cdot (-k\nabla T) = \Delta(kT)$$

where the last equality holds because we assume that the thermal conductivity k is constant for each fluid considered. This Laplacian term models the heat diffusion and, since it involves derivatives of the unknown T, is moved at the left hand side. The resulting equation for the energy conservation writes as

$$\frac{\partial(\rho c_p T)}{\partial t} + \nabla \cdot (\rho c_p T \mathbf{u}) - \Delta(kT) = 0$$

and describes only the transport and diffusion of the thermal energy. For the applications of our interest, also the heat exchange phenomena must be considered, such as conduction with the ground and convection and radiation heat losses with the environment. Conduction is implemented as a boundary condition (treated in Appendix A), while the radiative and convective phenomena take place at the free surface, namely at the interface between the two phases, and their implementation is described in the following two paragraphs under the assumption that the liquid phase is hotter than the gaseous phase. Moreover, we underline that, since we deal with a two phase model, we need to account the thermophysical characteristics of both fluids, liquid and gas, so we introduce the following parameters for both of them: the density ρ_l and ρ_g, the specific heat $c_{p,l}$ and $c_{p,g}$, the thermal conductivity k_l and k_g.

2.2.2.1 Radiative heat loss term

From the Stefan-Boltzmann law, the radiative heat loss per unit surface that takes place on the free-surface of the fluid is $\epsilon\sigma_{SB}f(T^4 - T_{env}^4)$, where ϵ is the emissivity, σ_{SB} is the Stefan-Boltzmann constant, f is the fractional area of the exposed inner core and T_{env} is the environmental temperature, as described in §2.1.5.4. As this phenomenon happens above the phase interface, for each control volume located at the interface of the two fluids, the radiative heat loss is also proportional to the free surface area A_{fs} related to such control volume (see §2.2.2.2 for details about the differences between the 3D model and the depth-averaged model). Since the energy equation is expressed per unit volume, then it is necessary also to divide for the volume of each control volume, denoted as Vol. Finally, the radiative contribution to add as source term to the energy equation is the term:

$$-\epsilon\sigma_{SB}f\frac{A_{fs}}{Vol}(T^4 - T_{env}^4).$$

2.2.2.2 Convective heat loss term

Convective heat loss is a transfer of heat referred to as the motion of a fluid. It takes place between the surface of a hot body or of hot fluid, and a colder fluid (liquid or gas) that is warmed from that, see §2.1.5.3 for a further description. For the applications of our interest, we try to model the natural situation of a hot fluid moving on a surface and exchanging heat with the environment. We would expect that the air in contact with the fluid gets warmer and starts a convective motion, so that new fresh air would get in contact with the fluid and little by little the fluid should cool down. Since we are adopting an incompressible assumption, this natural situation cannot be described exactly. In our case, even if the fluid warms the air because of the diffusion, there would never be a density variation of the air: the warm air would remain close to the surface, getting hotter and hotter, with the consequence that the fluid would not cool down as it naturally should. Since the real event cannot be captured by our basic model, we compensate for this disadvantage by adding a source term in the heat equation that accounts for the supposed heat loss due to convection. This term will be active only on the fluid surface, exactly as the radiative term.

The term describing the convective heat loss depends on the difference between the fluid and the environment temperatures, it is proportional to the heat transfer coefficient λ and to the fractional area of the exposed inner core f, as described in §2.1.5.3. As observed for the radiative term in the previous section §2.2.2.1, also for the convective heat loss term it is necessary to multiply for the area of the free surface A_{fs} and divide it by the volume Vol of the control volume. In summary, this term has the following expression:

$$-\lambda f\frac{A_{fs}}{Vol}(T - T_{env}).$$

There is an additional consideration to make: since the convective heat loss is fully described by this term, the heat diffusion between the two phases should not be present. So, the diffusion at the phase interface is suppressed. Considering that the function $\chi_\Sigma(\mathbf{x}, t)$ denotes the indicator function of the phase interface Σ, we use its complementary function $\bar{\chi}_\Sigma(\mathbf{x}, t)$ as a coefficient of the Laplacian term to override the diffusion at the phase interface. Finally, the complete energy equation is

$$\frac{\partial(\rho c_p T)}{\partial t} + \nabla\cdot(\rho c_p T\mathbf{u}) - \bar{\chi}_\Sigma\Delta(kT) = -\frac{\epsilon\sigma_{SB}fA_{fs}}{Vol}\left(T^4 - T_{env}^4\right) - \frac{\lambda fA_{fs}}{Vol}\left(T - T_{env}\right). \quad (2.58)$$

Further considerations

We compare the energy Eq. (2.58) of the 3D model with the temperature Eq. (2.42c) derived for the depth-averaged model to highlight differences and similarities.

- The greatest difference is given by conservative variables. In the shallow water model, it is hT $[m\,K]$, the depth-averaged temperature referred to the whole fluid thickness; in the 3D model, it is the thermal energy per unit volume $\rho c_p T$ $[kg\,m^{-1}s^{-2}]$.
- In the shallow water case, T is the averaged value over all the fluid depth; in the 3D model, T represents the temperature of a single fluid parcel that is usually a fraction of the entire depth.
- The temperature equation of the depth-averaged model is defined *per unit surface*, instead, the energy equation of the 3D model is defined *per unit volume*. Hence, in the shallow water model, the free surface area parameter A_{fs} (that instead is necessary in the 3D case) we simplified the radiative and convective terms. The example of radiative heat transfer term for the depth-averaged case (that was derived in §2.1.5.4) is shown:

$$-\frac{\epsilon\sigma_{SB} f \cancel{A_{fs}}}{\rho c_p \cancel{A_{fs}}}\left(T^4 - T_{env}^4\right) = -\frac{\epsilon\sigma_{SB} f}{\rho c_p}\left(T^4 - T_{env}^4\right).$$

- In the 3D case, also the heat diffusion phenomenon is modeled, whereas it is missing in the depth-averaged model. We could include it even in the shallow water model by adding an appropriate Laplacian term.
- In the 3D case, the heat conduction with the ground concerns only the boundary parcels of fluid that are at the ground, and is referred to as a boundary condition, whereas the radiative and convective terms only involve the interface between the phases. In the depth-averaged model, these three phenomena are instead accounted together and for every fluid parcel, in fact each fluid parcel for which we write the equation simultaneously includes the surface and the bottom of the fluid. Moreover, in the shallow water model, conduction with the ground cannot be prescribed as a boundary condition because the boundary, in such problems, is only constituted by the lateral sides of the domain.

2.2.3 The final system

To sum up, the equations that we are going to solve are:

$$\nabla \cdot \mathbf{u} = 0, \tag{2.59a}$$

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla \cdot \left(\rho \mathbf{u}\mathbf{u}^T\right) = -\nabla p + \nabla \cdot \boldsymbol{\tau} + \rho\mathbf{g} + \mathbf{f}_\Sigma, \tag{2.59b}$$

$$\partial_t \alpha + \nabla \cdot (\mathbf{u}\alpha) = 0, \tag{2.59c}$$

$$\frac{\partial(\rho c_p T)}{\partial t} + \nabla \cdot (\rho c_p T \mathbf{u}) - \bar{\chi}_\Sigma \Delta(kT) = -\frac{\epsilon\sigma_{SB} f A_{fs}}{Vol}\left(T^4 - T_{env}^4\right) - \frac{\lambda f A_{fs}}{Vol}\left(T - T_{env}\right), \tag{2.59d}$$

namely the continuity equation that under the assumption of constant density reduces to a kinematic constraint, the equation for the momentum conservation, the α-equation that permits to distinguish the two phases, and the energy equation that determines the temperature. We have found again a system of PDES which falls into the framework presented in §1.1.6.

2.3 Other models

In this section, we show some alternatives to our models which are present in the literature. In §2.3.1, we introduce some multiphase modeling approaches different from the VOF that we adopted for the 3D case. §2.3.2 is devoted to a review of the wide variety of possible approaches to modeling lava flows that are different from the depth-averaged and 3D models we employed.

2.3.1 Other multiphase models

A kind of method belonging to the group of the *interface-capturing methods* (see Figure 2.9b), alternative to VOF presented in §2.2.1, is the first appeared in literature. The **Marker-and-Cell** (MAC) family, introduced by Harlow and Welsh [115], is characterized by mass-less particles which are used as markers to locate the free surface; the mathematical model is enriched by adding specific equations involving these particles. According to the classification introduced in §1.1.3, this may be considered as a "semi-Lagrangian" method, since it couples the Lagrangian approach to model the markers with the Eulerian one of the other equations. MAC allows to deal with well complex phenomena like wave breaking or smoke diffusion, but on the other hand it is computationally very expensive because of the large number of particles to track which corresponds to a large number of other equations to add to the governing equations for the fluid flow, see Figure 2.11a.

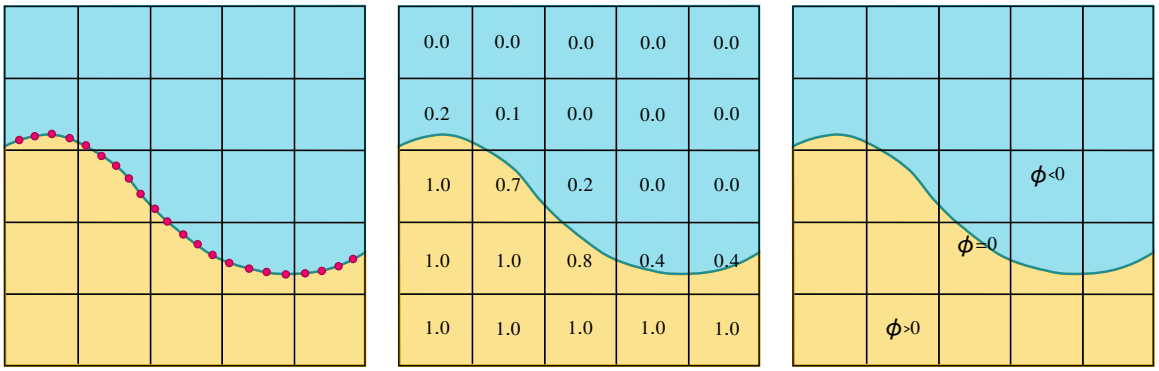
From a numerical point of view, even though the VOF approach is more efficient than the MAC method, the interface is usually smeared on one, two or three cells because of the difficulties of computing the advection without any diffusion. For this reason some numerical techniques that help to keep the interface sharp and compressed have been developed (as will be presented in §4.2).

Another family of interface-capturing method was proposed by Osher and Sethian [201] and it is based on the **level set** formulation. In this case a level set function ϕ is defined onto the computational domain and the interface is defined by the iso-surface of the value $\phi = 0$, while the function assumes on the other points a value expressing the distance with sign from the interface. The two phases are simply associated with the sign of the level set function, see Figure 2.11c, and the evolution of the interface is computed by solving a transport equation for the level set function

$$\partial_t \phi + \nabla \cdot (\mathbf{u}\phi) = 0.$$

A great advantage of this method is that the function ϕ describing the interface is smooth, whereas in the case of the VOF method the function α has a discontinuity exactly at the interface, therefore we don't find here the same difficulties to maintain the discontinuity. On the other side, from a numerical point of view, when the level set function becomes

too complicated (namely when the norm of the gradient of ϕ is far from the unity) then it is reinitialized as the distance function from the interface, in order to allow to keep the norm of the gradient close to the unity (avoiding ill-conditioning), see Min [193] for more details. Moreover, the level set function is never involved in any conservation equation therefore the level set method does not maintain exactly the mass unless doing some modifications.



(a) *Marker-and-Cell.* (b) *Volume of Fluid.* (c) *Level Set.*

Figure 2.11: Methods belonging to the interface-capturing group.

2.3.2 Other existing models for lava flows

In this section, we provide a short overview of the existing models for lava flow simulations (neglecting the depth-averaged and 3D models that we have treated in details). Models might be classified as deterministic or stochastic, or according with the numerical approach they are based on, as done in [52, 142].

Channeled models. Channeled models consider confined fluids that move in only one direction, down-slope, unlike from the free flows that can spread also transversely. Since this restriction simplifies the physical processes related to the fluid emplacement, this model often treats complex thermal and rheological models. Also, channeled models are used to study the behavior of fluids, for example, this model was employed in Lev and James [164] to examine the influence of cross-sectional channel geometry on the rheology and on the velocity and temperature profiles.

The main code implementing this approach is FLOWGO [116, 118, 119], which models the dynamics of lava in a channel by considering the radiative and convective thermal exchanges with the environment and the conductive one with the soil. The cooling rate that results is used to determine a crystallization rate that affects the rheology as much as the temperature directly. Codes of channeled models are fast to execute because do not compute the fluid motion. Nevertheless, a model disadvantage is that it imposes restrictions on the choice of the channel dimensions at the vent in order to match the effusion rate inputs and, as already stated, the vent conditions influence the flow results [270].

Cellular automata. The cellular automata approach is a popular 2D model. The computational domain is discretized by a 2D grid of cells and each cell has attributes such as altitude, fluid thickness, and fluid temperature. Dynamics is described through

the variations of the properties of the cells with time and, for each cell, such variations depend on the state of the neighboring cells. Complex thermal and viscosity models are implemented. This model, at first, used the plastic rheology as a stop condition for the flow, whereas, nowadays the stop criteria is determined by a solidification temperature, defined a priori. Moreover, the model allows the solidified lava to become part of the topography simply updating the altitude registered as a property of the cells.

Each code that implements cellular automata models has a different strategy for the mass, momentum, and energy transfer between cells. Cellular automata codes are relatively fast to run and some of the most known codes are MAGFLOW [17, 68, 98, 121, 263], SCIARA [5, 7, 8, 60, 61, 62], FLOWFRONT [266, 280], and MOLASSES [49, 70, 155].

Nuclear-based models. There is a family of models for lava flows that was initially born to model another event where the cooling of a hot fluid spreading on the topography is observed: this family originates in the context of nuclear reactor safety. The accidents of Three Mile Island, in 1979, and Chernobyl, in 1986, are nuclear reactor disasters that induced the nuclear safety groups to control any potential accident in the reactor containment in order to minimize off-site consequences. During a severe nuclear reactor meltdown, it is fundamental that the fusion melt product (corium) quickly spread onto the widest possible surface to extinguish it as fast as possible. Since this event is definitely similar to the lava flow dynamics, the codes developed for the nuclear disaster simulations can be used also by volcanologists.

Codes that implement this nuclear model are CORFLOW [275], CROCO [192], LAVA [199], MELTSPREAD [82], RASPLAV [20], SAMPSON [124] and THEMA [247]. Even though these codes implement a modified version of the cellular automata models in which also the height is discretized with several cells (producing a multilayer model), there is not the direct modeling of the free surface. The only further difference between nuclear and lava models is the absence of the topography, hence its implementation is required to convert these nuclear engineering codes to lava flow emplacement models. For example, the code LavaSIM [124, 125, 228] was adapted to lava flow simulations [94].

Meshless and 'bottom-up' methods. The approaches described so far have the flow model that depends on the topography, and this may be an expensive cause when the computations require numerous re-meshing procedures. So, meshless methods may be less expensive in terms of computational cost. Furthermore, the approaches described previously are based on the so-called 'top-down' approach, in which a macroscopic system is modeled by a system of PDEs. On the opposite, the meshless methods are built according to a 'bottom-up' approach based on the description of microscopic systems. The fluid is not seen as a continuum (so the Navier-Stokes equations are not adopted) but composed of particles, and collision methods are employed to model their interactions and hence to derive the dynamics. The Lattice Boltzmann methods (LBM) [205, 206] and smoothed particles hydrodynamics (SPH) [122] are the most important approaches to the bottom-up models.

Bottom-up methods are not so widely used in geosciences and only the SPH approach, which is naturally 3D, has been applied to the lava flow simulations: an example of application is presented in Herault et al. [122], others are in Bilotta et al. [18], Zago et al. [281].

Stochastic models. The stochastic modeling approach produces probability distributions of the final fluid emplacement, instead of a unique solution. Such an approach requires that a deterministic code runs several times with small changes in the parameters in order to build the probability distribution. Even though in principle most codes can be used for this application, only codes that are very fast to execute are used in practice. The fastest codes for fluid flows correspond to the most simplified models for which the physical modeling is strongly simplified and the fluid is described as a gravity current that moves following the topography steepest slope [72]. In the context of lava flows, this also means that the viscosity and thermodynamic models may not be accounted for. A drawback of the most simplified codes is the absence of stopping criteria for the flow advance; furthermore, they have no description of the temporal evolution. Those codes are used to create hazard flow maps. On the contrary, when using more accurate models, e.g. the cellular automata-based codes, also the time evolution of the front propagation is assessed.

DOWNFLOW is a probabilistic code for lava flow emplacement [83, 252] adopted also recently to produce lava hazard maps [39, 254]. Even the code ELFM "Etna Lava Flow Model" [64] is used lately for lava hazard maps [56, 246]. VORIS "Volcanic Risk Information System" [84] is a topography-based code that has been also included in VOLCANBOX [187] (a simple method for assessing volcanic hazards and risks). Other codes have been developed, as LASZLO "lava flow zonation system using low-cost methods" [21], models for random pahoehoe lava-lobe emplacement [103, 112], and the recent MULTIFLOW [230].

Selection of the best code. The selection of a modeling approach and of a code must be driven by the objectives to reach. As seen so far, the complexity described by the physical model and the code execution speed are two aspects inversely proportional to take into account, therefore the choice to do is usually a compromise between accuracy and speed. On one hand, the stochastic codes have the most simplified physics to produce faster results and can be of great support during an effusive event to forecast rapidly a possible scenario according to the actual parameters derived from the direct observations of the eruption. On the other hand, these models cannot describe the entire complexity of the physical processes that, in turn, influence the dynamics. The deterministic codes that take into account viscosity and thermodynamics, being relatively slower, are mostly used in the phases before and post-eruption. To conclude, the *best* code is the one that satisfies the current necessity of the user.

Chapter 3

Numerical discretization of the depth-averaged model

Due to the non-linearities present in the shallow water equations, it is possible to find analytical solutions only in particular cases, and in general, a numerical approach is adopted for their solution. When dealing with the numerical solution of depth-averaged flow models, an important issue is properly tracking wet/dry fronts, i.e., the propagation of interfaces between regions occupied by the flow (wet) and regions without flow (dry). A large number of numerical models solving for the shallow water equations are based on explicit finite volume schemes [28, 85, 157], and they adopt different strategies to deal with wetting and drying. It is important to remark that the stability of any explicit scheme is controlled by the Courant-Friedrichs-Lewy (CFL) condition (introduced and described in the sections §1.2.2.2 and §1.2.9) and by the maximum characteristic speed of the governing equations, which depends on the square root of flow thickness. For this reason, a numerical scheme producing negative thickness is not only physically unrealistic but also results in numerical problems. A simple but effective method to capture wet/dry fronts and to overcome these problems would be set to zero any negative thickness produced by a numerical scheme, together with the corresponding velocities [14], but we observe that this approach does not preserve the total mass of the system. In Godunov-type schemes, other conservative methods have been proposed to treat wetting and drying [10, 170]. Our approach for the spatial discretization is based on the KNP scheme described in section §1.2.8, a central-upwind finite-volume scheme. On the one hand, such an approach guarantees the positivity of the solution, and on the other hand, it is able to capture steady states (see Eq. (2.43) over complex topography, like those represented by a null velocity and a flat free surface (like a lake at rest). In literature, methods able to preserve these stationary states are generally called well-balanced methods [23]. However, in addition to the numerical issues described above, computational instabilities may also arise in the numerical simulation of wet/dry fronts when friction terms inversely proportional to flow thickness are considered, like, for example, in the Manning formulation [183]. For this reason, it is essential to employ appropriate numerical schemes for time discretization, such as semi-implicit schemes [10, 14], or implicit-explicit Runge-Kutta methods [237, 238]. In this work, we propose a combination of some of the above-mentioned existing techniques. The spatial discretization is performed using a modified version of the central-upwind finite volume scheme introduced by Kurganov and Petrova [158]. They developed a numerical method to solve the classic shallow water equations basing on similar ideas to those of the general KNP scheme treated in §1.2.8. We adapted such method to our

depth-averaged modified model, derived in §2.1, proving that important properties like **well-balancing** and **positivity-preserving** for near-dry states are satisfied, see section §3.1. We couple this approach with an implicit-explicit Runge-Kutta technique employed to the temporal discretization (proposed in [238] for the classical shallow water equations, but not considered in [158]), where a suitable selection of the stiff terms to be treated implicitly is performed in section §3.2. Clever use of complex arithmetic, described in section §3.3, ensures an accurate computation of Jacobians, which is a necessary requirement for the efficient solution of the implicit part of the scheme. That is a general technique borrowed from [248] and has not yet been considered in the literature cited above. The present combination of choices is new and appears quite efficient. The conservative and positivity-preserving explicit central-upwind numerical scheme allows for robust and accurate tracking of flow fronts. Meanwhile, the implicit treatment of nonlinear viscous terms allows us to properly simulate steady-state equilibrium states and flow-stopping conditions without the need for any ad-hoc empirical criteria. In §3.4 several numerical simulations obtained with our modified scheme are presented.

Vectorial notation

We have to solve the system of Eqs. (2.40) or the system (2.42) and, for the sake of simplicity, we present the numerical scheme for the one dimensional case. We already gave some hints and details about the numerical discretization of a multidimensional system in section §1.2.9 and for additional considerations the reader might consult the discretization of the 2D version of a different shallow water model in de' Michieli Vitturi et al. [67]. We rewrite here the 1D version of the system (2.42)

$$\begin{aligned}
&\frac{\partial(\rho h)}{\partial t} + \frac{\partial(\rho h U)}{\partial x} = \rho R,\\
&\frac{\partial(\rho h U)}{\partial t} + \frac{\partial}{\partial x}\left(\beta_u \rho h U^2\right) + \frac{\partial}{\partial x}\left(\frac{1}{2}\rho g h^2\right) = -\rho g h \frac{\partial}{\partial x} B - \rho\gamma U,\\
&\frac{\partial(hT)}{\partial t} + \frac{\partial}{\partial x}\Big[\big(\beta_T T + (1-\beta_T)\, T_{soil}\big) hU\Big] = -\mathcal{H}\big(\zeta - \psi\big)\big(T - T_{soil}\big)\\
&\qquad -\mathcal{W}\big[\zeta T + (1-\zeta) T_{soil} - T_{env}\big] - \mathcal{E}\Big[\big(\zeta T + (1-\zeta) T_{soil}\big)^4 - T_{env}^4\Big]\\
&\qquad + \mathcal{K} U^2 \exp\big[-b(T - T_{ref})\big] + R T_{vent},
\end{aligned} \tag{3.1}$$

that is coupled with the state equation for the temperature dependent density

$$\rho = mT + \rho_0.$$

Naming $\mathbf{q}$ the vector of conservative variables, we denote by $\mathbf{f}(\mathbf{q})$ the vector of the advective fluxes:

$$\mathbf{q} = \begin{bmatrix} \rho h \\ \rho h U \\ h T \end{bmatrix}, \quad \mathbf{f}(\mathbf{q}) = \begin{bmatrix} \rho h U \\ \beta_u \rho h U^2 + \dfrac{\rho g}{2} h^2 \\ \big(\beta_T T + (1-\beta_T) T_*\big) h U \end{bmatrix},$$

where T_* might represent T_{surf} or T_{soil}, depending on which system is considered. The source terms of the equations are divided in two vectors $\mathbf{E}(\mathbf{q})$ and $\mathbf{S}(\mathbf{q})$ so that we can write the 1D system in the vectorial form:

$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{q})}{\partial x} = \mathbf{E}(\mathbf{q}) + \mathbf{S}(\mathbf{q}). \tag{3.2}$$

When the first system of Eq. (2.40) is considered, the vectors of the source terms have respectively the topography and the viscosity terms:

$$\mathbf{E}(\mathbf{q}) = \begin{bmatrix} 0 \\ -\rho g h B_x \\ 0 \end{bmatrix}, \quad \mathbf{S}(\mathbf{q}) = \begin{bmatrix} 0 \\ -\rho\gamma U \\ 0 \end{bmatrix};$$

instead, if the second system of Eq. (2.42) is adopted, the vectors contain also the viscous heating term, the effusion rate terms, and the conductive, convective and radiative heat exchanges terms, respectively:

$$\mathbf{E}(\mathbf{q}) = \begin{bmatrix} \rho R \\ -\rho g h B_x \\ \mathcal{K}\exp\big(-b(T-T_{ref})\big) + RT_{vent} \end{bmatrix}, \quad \mathbf{S}(\mathbf{q}) = \begin{bmatrix} 0 \\ -\rho\gamma U \\ \bigstar \end{bmatrix}$$

where $\bigstar$ stands for

$$-\mathcal{H}\big(\zeta-\psi\big)\big(T-T_{soil}\big) - \mathcal{W}\big[\zeta T + (1-\zeta)T_{soil} - T_{env}\big] - \mathcal{E}\big[\big(\zeta T + (1-\zeta)T_{soil}\big)^4 - T_{env}^4\big].$$

The distinction between the two source terms will be exploited in §3.2.

3.1 Spatial discretization

In order to apply a Finite Volume scheme, according to §1.2, the domain is divided into a uniform grid of width Δx, and for each cell we denote by x_i its midpoint and indicate by $C_i = \left[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}\right]$ the i-th cell. An important feature of a numerical scheme is a proper treatment of the topography. Here, as a preliminary step, the original topography $B(x)$ is approximated by a continuous linear-piecewise function $\widetilde{B}(x)$, as shown in Figure 3.1, whose values $\widetilde{B}_{i+\frac{1}{2}}$ at the interfaces $x_{i+\frac{1}{2}}$ are obtained by averaging the limits of $B(x)$ from both the sides

$$\widetilde{B}_{i+\frac{1}{2}} := \frac{B(x_{i+\frac{1}{2}})^+ + B(x_{i+\frac{1}{2}})^-}{2}, \qquad \text{with } B(x_{i+\frac{1}{2}})^\pm := \lim_{\delta x \to 0^\pm} B(x_{i+\frac{1}{2}} + \delta x).$$

The approximation $\widetilde{B}$ and its derivative are used for the finite-volume numerical discretization in the cell C_i of the second component of the source term $\mathbf{E}(\mathbf{q})$:

$$\frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} (-\rho g h B_x)dx \approx -g\rho_i h_i \frac{\widetilde{B}_{i+\frac{1}{2}} - \widetilde{B}_{i-\frac{1}{2}}}{\Delta x}. \tag{3.3}$$

We denote by $\mathbf{Q}_i$ the approximation of the average of $\mathbf{q}(\cdot,t)$ over the i-th cell

$$\mathbf{Q}_i(t) \approx \frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathbf{q}(x,t)dx \ ,$$

then we apply a semi-discretization of Eqs. (3.2) obtaining the following system of ODEs in each cell:

$$\frac{d}{dt}\mathbf{Q}_i(t) = -\frac{\mathbf{F}_{i+\frac{1}{2}}(t) - \mathbf{F}_{i-\frac{1}{2}}(t)}{\Delta x} + \mathbf{E}_i(t) + \mathbf{S}_i(t) \tag{3.4}$$

where $\mathbf{F}_{i\pm\frac{1}{2}}$ are the numerical fluxes, here defined accordingly with Kurganov and Petrova [158] (KP in the following). Our modified version of the central-upwind finite volume KP scheme is briefly summarized in the following steps (see the original reference [158] for details).

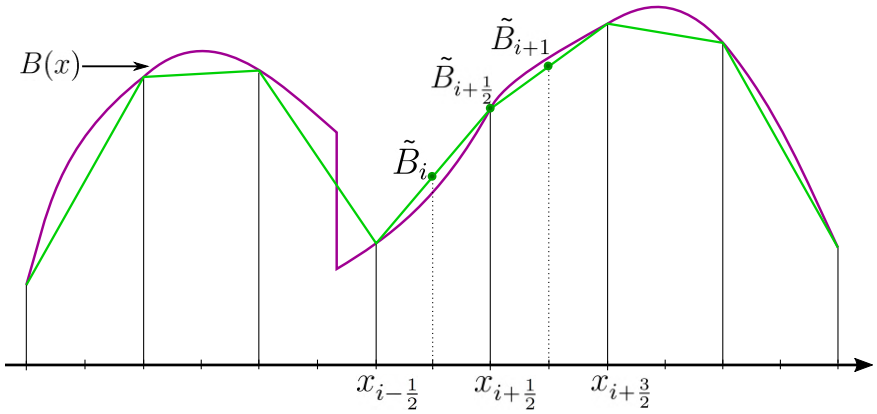


Figure 3.1: *Piecewise linear approximation of the bottom.* The *purple* line represents the bottom topography function $B(x)$, the *green* line shows its continuous piecewise linear approximation $\tilde{B}(x)$.

(1) Solution reconstruction. From the values of the conservative variables $\mathbf{Q}$, we obtain an additional set of variables $\mathbf{P} = [w, U, T]^T$ (where $w := h + B$ denotes the free-surface height), by using the following relations

$$h = \frac{\mathbf{Q}^{(1)} - m\mathbf{Q}^{(3)}}{\rho_0}, \quad T = \frac{\mathbf{Q}^{(3)}}{h}, \quad \rho = mT + \rho_0, \quad U = \frac{\mathbf{Q}^{(2)}}{\rho h}$$

($\mathbf{Q}^{(j)}$ denotes the j-th component of $\mathbf{Q}$) and we use their discretized values in most of our computations, differently from [158]. When the height h is very small, an appropriate de-singularization procedure is necessary, in order to prevent inaccuracies in velocity and temperature values. To do that, we adopt the following expressions to compute the temperature and velocity values instead of the previous ones

$$T = \frac{\sqrt{2}h\mathbf{Q}^{(3)}}{\sqrt{h^4 + \max(h^4, \varepsilon)}}, \quad U = \frac{\sqrt{2}h\mathbf{Q}^{(2)}}{\rho\sqrt{h^4 + \max(h^4, \varepsilon)}}$$

where ε is a small a-priori chosen positive number, according to the analogous formulas suggested in [158] for the set of conservative variables.

By using the cell values $\{\mathbf{P}_i\}_i$ calculated from the cell averages $\{\mathbf{Q}_i\}_i$, we define a discontinuous piecewise linear reconstruction over each cell:

$$\widetilde{\mathbf{P}}_i(x) = \mathbf{P}_i + \boldsymbol{\sigma}_i(x - x_i), \quad x_{i-\frac{1}{2}} < x < x_{i+\frac{1}{2}}, \quad \forall i, \tag{3.5}$$

where the slopes $\{\boldsymbol{\sigma}_i\}_i$ are computed by using a suitable geometric limiter. One choice of slope (that gives second-order accuracy for smooth solutions) is given by the *minmod limiter*, defined in Eq. (1.92), applied to each component. An other example of choice for the slope is offered by the *generalized minmod limiter*, defined as

$$\boldsymbol{\sigma}_i = \text{minmod}\left(\theta\frac{\mathbf{Q}_i - \mathbf{Q}_{i-1}}{\Delta x}, \frac{\mathbf{Q}^n_{i+1} - \mathbf{Q}_{i-1}}{2\Delta x}, \theta\frac{\mathbf{Q}_{i+1} - \mathbf{Q}_i}{\Delta x}\right), \tag{3.6}$$

where θ is a suitable parameter and the *minmod* function is defined as follows:

$$\text{minmod}\,(a_1, a_2, \dots) = \begin{cases} \min_j\{a_j\}, & \text{if} \quad a_j > 0 \quad \forall j, \\ \max_j\{a_j\}, & \text{if} \quad a_j < 0 \quad \forall j, \\ 0, & \text{otherwise.} \end{cases} \tag{3.7}$$

The reconstructed solution might be discontinuous at the interfaces points $x_{i+\frac{1}{2}}$, then we define $\mathbf{P}^L_{i+\frac{1}{2}}$ and $\mathbf{P}^R_{i+\frac{1}{2}}$ as the left and right side values of the reconstructed solution respectively, represented in Figure 3.2

$$\mathbf{P}^L_{i+\frac{1}{2}} := \mathbf{P}_i + \frac{\Delta x}{2}\boldsymbol{\sigma}_i\,, \qquad \mathbf{P}^R_{i+\frac{1}{2}} := \mathbf{P}_{i+1} - \frac{\Delta x}{2}\boldsymbol{\sigma}_{i+1}.$$

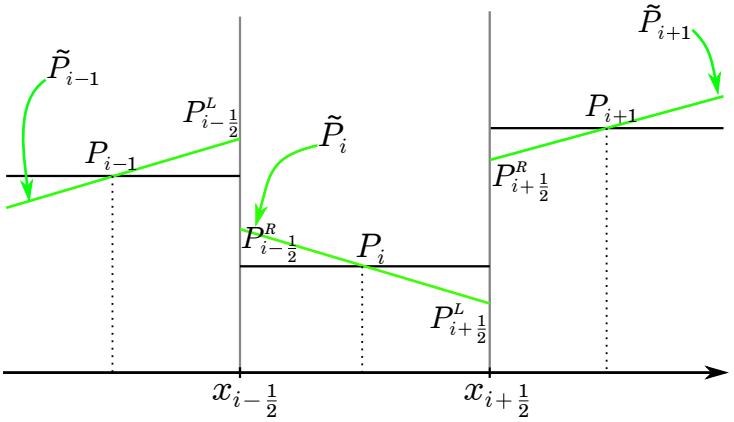


Figure 3.2: *Solution reconstruction.* The *black* lines represent the piecewise constant reconstruction of the $\mathbf{P}_i$ variables, the *green* lines depict the linear reconstructions $\{\widetilde{\mathbf{P}}_i\}_i$.

In addition, a suitable correction is needed for the reconstructed values $w^{L/R}_{i+\frac{1}{2}}$ of the first components, in order to preserve the positivity of each $h^{L/R}_{i+\frac{1}{2}} := w^{L/R}_{i+\frac{1}{2}} - \widetilde{B}_{i+\frac{1}{2}}$, not guaranteed by the geometric limiters. In order to give the idea about how such problem may rise, one considers the situation represented in Figure 3.3 where the *generalized minmod limiter* (3.6)) is applied for the linear reconstruction of w. In such configuration, despite the adoption of a geometric limiter, the reconstruction in the i-th cell produces that $w^L_{i+\frac{1}{2}} < B_{i+\frac{1}{2}}$, hence there is a negative value of the thickness $h^L_{i+\frac{1}{2}} < 0$, which is physically unacceptable.
A correction to the reconstruction of w is necessary in two cases:

- if $w^L_{i+\frac{1}{2}} < \widetilde{B}_{i+\frac{1}{2}}$,
- if $w^R_{i-\frac{1}{2}} < \widetilde{B}_{i-\frac{1}{2}}$.

The first case is portrayed in Figure 3.4, and when this situation occurs, then the value of $w^L_{i+\frac{1}{2}}$ is set equal to the topography, the slope of the linear reconstruction is updated using the cell center value and the corrected value at the interface, while $w^R_{i-\frac{1}{2}}$ is obtained computing the linear reconstruction in the i-th cell:

$$\begin{aligned} w^L_{i+\frac{1}{2}} = \widetilde{B}_{i+\frac{1}{2}} \implies w^R_{i-\frac{1}{2}} &= \frac{w^L_{i+\frac{1}{2}} - w_i}{\Delta x/2}(x_{i-\frac{1}{2}} - x_{i+\frac{1}{2}}) + \widetilde{B}_{i+\frac{1}{2}} \\ &= -\frac{\widetilde{B}_{i+\frac{1}{2}} - w_i}{\Delta x/2}\Delta x + \widetilde{B}_{i+\frac{1}{2}} \\ &= 2w_i - \widetilde{B}_{i+\frac{1}{2}}. \end{aligned}$$

The other case is corrected analogously, so we resume the two procedures as:

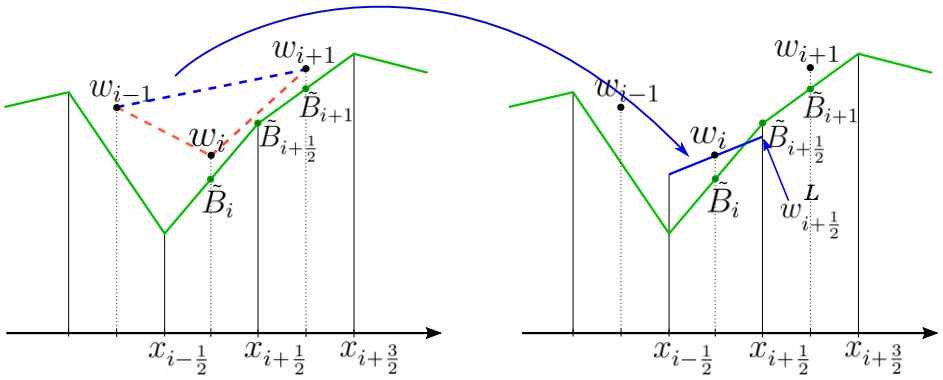


Figure 3.3: *Negative thickness.* The *green lines* represent the piece-wise linear continuous reconstruction of the bottom $\widetilde{B}(x)$ and the black dots are the cell values of $w = B + h$. *Left:* the dashed lines are the three slopes that are compared from the *generalized minmod limiter* (3.6) to determine the slope to use for the reconstruction over the i-th cell; the blue line refers to the slope effectively accounted by the limiter. *Right:* The blue line is the reconstruction over the i-th cell with the limiter: the reconstructed value at the interface $w^L_{i+\frac{1}{2}}$ produces the negative value of the thickness $h^L_{i+\frac{1}{2}}$.

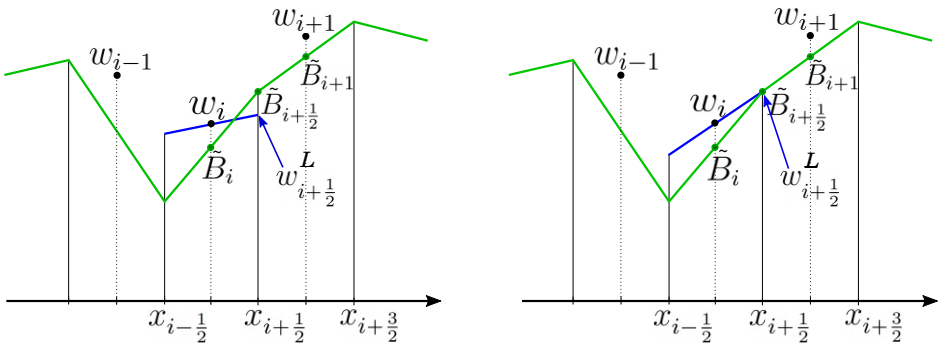


Figure 3.4: *Correction of h.* The *green lines* represent the piece-wise linear continuous reconstruction of the bottom $\widetilde{B}(x)$, the *black dots* are the cell values of $w = B + h$ and the *blue lines* are the linear reconstruction of w in the i-th cell. *Left:* the current reconstruction of w shows a problematic value of $w^L_{i+\frac{1}{2}}$ that must be corrected. *Right:* the reconstruction of w was corrected.

- if $w^L_{i+\frac{1}{2}} < \widetilde{B}_{i+\frac{1}{2}}$, then set $w^L_{i+\frac{1}{2}} = \widetilde{B}_{i+\frac{1}{2}} \implies w^R_{i-\frac{1}{2}} = 2w_i - \widetilde{B}_{i+\frac{1}{2}}$;
- if $w^R_{i-\frac{1}{2}} < \widetilde{B}_{i-\frac{1}{2}}$, then set $w^R_{i-\frac{1}{2}} = \widetilde{B}_{i-\frac{1}{2}} \implies w^L_{i+\frac{1}{2}} = 2w_i - \widetilde{B}_{i-\frac{1}{2}}$.

The solution reconstruction is a key point to guarantee the well-balancing property of a numerical scheme. In §3.1.1, we show that the method introduced here continues to guarantee this property, even if the equations are modified and the linear reconstruction is computed for a different set of variables with respect to [158].

(2) Values of the conservative variables at the interfaces. From the values $\mathbf{P}^L_{i+\frac{1}{2}}$ and $\mathbf{P}^R_{i+\frac{1}{2}}$ of our variables at the interfaces, we compute $\mathbf{Q}^L_{i+\frac{1}{2}}$ and $\mathbf{Q}^R_{i+\frac{1}{2}}$, namely the values of the conservative variables at the interfaces:

$$\mathbf{Q}^{(1)} = \underbrace{\left(m\mathbf{P}^{(3)} + \rho_0\right)}_{\rho}\underbrace{\left(\mathbf{P}^{(1)} - B\right)}_{h}, \quad \mathbf{Q}^{(2)} = \mathbf{Q}^{(1)}\mathbf{P}^{(2)}, \quad \mathbf{Q}^{(3)} = \left(\mathbf{P}^{(1)} - B\right)\mathbf{P}^{(3)}.$$

(3) Estimation of the local speed through the interfaces. The reconstructed solution propagates with right-sided and left-sided local speeds $a^{\pm}_{i+\frac{1}{2}}$, which are estimated in terms of the eigenvalues of the Jacobian of $\mathbf{f}$. We denote the Jacobian as $\mathbf{f}'$ and it is as follows:

$$\mathbf{f}'(\mathbf{q}) = \begin{bmatrix} 0 & 1 & 0 \\ -\beta_u U^2 + gh & 2\beta_u U & 0 \\ -\beta_T UT/\rho & \beta_T T/\rho + (1-\beta_T)T_*/\rho & \beta_T U \end{bmatrix}.$$

This Jacobian has three eigenvalues, that are the elements of this spectrum:

$$Sp\,(\mathbf{f}') = \left\{ \beta_u\, U \pm \sqrt{\beta_u(\beta_u - 1)U^2 + gh},\ \beta_T U \right\}. \tag{3.8}$$

The velocities of propagation of the reconstructed solution at the interfaces depend on the maximum and minimum eigenvalues of the Jacobian evaluated at $\mathbf{Q}^{R/L}_{i+\frac{1}{2}}$:

$$\begin{aligned} a^{+}_{i+1/2} &= \max\left\{ \lambda_{\max}\left(\mathbf{f}'\left(\mathbf{Q}^{R}_{i+\frac{1}{2}}\right)\right), \lambda_{\max}\left(\mathbf{f}'\left(\mathbf{Q}^{L}_{i+\frac{1}{2}}\right)\right), 0 \right\}, \\ a^{-}_{i+1/2} &= \min\left\{ \lambda_{\min}\left(\mathbf{f}'\left(\mathbf{Q}^{R}_{i+\frac{1}{2}}\right)\right), \lambda_{\min}\left(\mathbf{f}'\left(\mathbf{Q}^{L}_{i+\frac{1}{2}}\right)\right), 0 \right\}. \end{aligned} \tag{3.9}$$

The computation of U is requested also here, but in this occasion the de-singularization is not necessary: we did it previously passing to the set of variables $\mathbf{P}$, with those values we have done the linear reconstruction, and since we have used geometric limiters, then all the velocities are limited at the interfaces.

(4) Computation of the numerical flux. We adopt the numerical flux expression used in [158], that is:

$$\mathbf{F}_{i+\frac{1}{2}}(t) = \frac{a^{+}_{i+\frac{1}{2}}\mathbf{f}\left(\mathbf{Q}^{L}_{i+\frac{1}{2}}\right) - a^{-}_{i+\frac{1}{2}}\mathbf{f}\left(\mathbf{Q}^{R}_{i+\frac{1}{2}}\right)}{a^{+}_{i+\frac{1}{2}} - a^{-}_{i+\frac{1}{2}}} + \frac{a^{+}_{i+\frac{1}{2}}a^{-}_{i+\frac{1}{2}}}{a^{+}_{i+\frac{1}{2}} - a^{-}_{i+\frac{1}{2}}}\left[\mathbf{Q}^{R}_{i+\frac{1}{2}} - \mathbf{Q}^{L}_{i+\frac{1}{2}}\right] \tag{3.10}$$

where the first term on the right side is an average of the evaluated fluxes weighted with the speeds of propagation, because the total flux at the interface depends both on the left-going and on the right-going fluxes, each weighted with the maximum speed in that direction. Notice that the time dependence of the right term, although it is omitted explicitly, belongs to both $a^{\pm}_{i+\frac{1}{2}}$ and $\mathbf{Q}^{R/L}_{i+\frac{1}{2}}$.

Summarizing the differences with respect to [158], the model equations (3.1) we refer to are richer, since we have an additional equation for the temperature transport, corrector coefficients β_u and β_T inside the flux terms, more source terms and also a variable density. From the numerical point of view, a major difference between our scheme and the original KP scheme is that we pass forward and backward to the set of variables $\mathbf{P}$ to compute the solution reconstruction of the conservative variables at the interfaces. Moreover, the presence of the corrector coefficients β_u and β_T impacts over the scheme by affecting the local speed of propagation.

3.1.1 Well-balancing property

As stated before, the modifications introduced here to the numerical scheme presented in [158] continue to guarantee the well-balancing property for stationary steady states. We provide a sketch of proof of this property.

One recalls that the stationary steady-state solutions of our system consist of null velocity and constant conditions for both free surface height, temperature and density (see equations (2.43)). In such cases, from the system of equations (3.2) we obtain:

$$\partial_x\left(\frac{1}{2}\rho g h^2\right) = -\rho g h B_x. \tag{3.11}$$

By proving that the equality (3.11) is preserved also by the numerical discretization fo the two terms, we confirm the well-balancing of the numerical scheme.

The flux term on the left side is discretized with the numerical fluxes $\mathbf{F}^{(2)}_{i-\frac{1}{2}}$ and $\mathbf{F}^{(2)}_{i+\frac{1}{2}}$ defined in Eq. (3.10). Since velocity is null, the conservative variable $\mathbf{Q}^{(2)} = \rho h U$ is null too, therefore the numerical flux at the right interface of the cell i reduces to:

$$\mathbf{F}^{(2)}_{i+\frac{1}{2}} = \frac{a^+_{i+\frac{1}{2}}\mathbf{f}^{(2)}\left(\mathbf{Q}^L_{i+\frac{1}{2}}\right) - a^-_{i+\frac{1}{2}}\mathbf{f}^{(2)}\left(\mathbf{Q}^R_{i+\frac{1}{2}}\right)}{a^+_{i+\frac{1}{2}} - a^-_{i+\frac{1}{2}}}, \tag{3.12}$$

where the flux values at the sides of the interfaces are given by $\mathbf{f}^{(2)}\left(\mathbf{Q}^{L/R}_{i+\frac{1}{2}}\right) = \frac{1}{2}\rho^{L/R}_{i+\frac{1}{2}} g\left(h^{L/R}_{i+\frac{1}{2}}\right)^2$. According to the numerical scheme, the linear reconstruction is applied to the free surface $w = h + B$ to find its interface values $w^L_{i+\frac{1}{2}}$ and $w^R_{i+\frac{1}{2}}$, from which we derive the interface values of h as

$$h^L_{i+\frac{1}{2}} = w^L_{i+\frac{1}{2}} - \widetilde{B}_{i+\frac{1}{2}}, \quad h^R_{i+\frac{1}{2}} = w^R_{i+\frac{1}{2}} - \widetilde{B}_{i+\frac{1}{2}}.$$

As the total thickness $w = h + B$ is constant, its reconstructed values on the left and right sides of each interface are equal, hence the same condition descends for h:

$$w^L_{i+\frac{1}{2}} = w^R_{i+\frac{1}{2}} \Rightarrow h^L_{i+\frac{1}{2}} = h^R_{i+\frac{1}{2}}.$$

From the assumption of constant density, we also have $\rho^L_{i+\frac{1}{2}} = \rho^R_{i+\frac{1}{2}} = \rho$ and from now we neglect the L/R notation for the interface values of h and ρ. According with these considerations, the numerical flux simplifies its expression

$$\mathbf{F}^{(2)}_{i+\frac{1}{2}} = \frac{a^+_{i+\frac{1}{2}}\left[\frac{1}{2}\rho g\left(h_{i+\frac{1}{2}}\right)^2\right] - a^-_{i+\frac{1}{2}}\left[\frac{1}{2}\rho g\left(h_{i+\frac{1}{2}}\right)^2\right]}{a^+_{i+\frac{1}{2}} - a^-_{i+\frac{1}{2}}} = \frac{1}{2}\rho g\left(h_{i+\frac{1}{2}}\right)^2$$

and the complete discretization of the flux term on the LHS of the stationary state Eq. (3.11) becomes

$$\frac{\mathbf{F}^{(2)}_{i+\frac{1}{2}} - \mathbf{F}^{(2)}_{i-\frac{1}{2}}}{\Delta x} = \frac{1}{2\Delta x}\rho g\left[\left(h_{i+\frac{1}{2}}\right)^2 - \left(h_{i-\frac{1}{2}}\right)^2\right] = \rho g h_i \frac{h_{i+\frac{1}{2}} - h_{i-\frac{1}{2}}}{\Delta x}, \tag{3.13}$$

where, because of the linearity of the interface reconstruction, h_i is the average of the interface values.

We observe that the discretization of the right term of Eq. (3.11), given by Eq. (3.3), can be written in the following equivalent way

$$-g\rho_i h_i \frac{\widetilde{B}_{i+\frac{1}{2}} - \widetilde{B}_{i-\frac{1}{2}}}{\Delta x} = -g\rho h_i \frac{\left(w_{i+\frac{1}{2}} - h_{i+\frac{1}{2}}\right) - \left(w_{i-\frac{1}{2}} - h_{i-\frac{1}{2}}\right)}{\Delta x}. \tag{3.14}$$

Because of the constant condition on the free surface, we have that $w_{i-\frac{1}{2}} = w_{i+\frac{1}{2}}$ and, by canceling these terms in the previous equation, we find the same expression as in Eq. (3.13), so we conclude the proof.

3.1.2 Positivity-preserving property

Kurganov and Petrova [158] proved that, for their model, an explicit temporal discretization of the central-upwind scheme with an appropriate choice of the time step, is not only well-balanced, but also positivity preserving for h. Despite the differences from them, for our modified scheme their result is still valid, but with an additional condition over the β coefficients in the flux terms of Eq. (3.1).

We premise in a specific lemma the nontrivial adjustments to the proof of the corresponding result in [158].

Lemma 3.1. *If the condition* $\max(\beta_u, \beta_T) \geqslant 1$ *is verified, then for each interface the following inequalities hold:*

$$(a) \quad a^-_{i+\frac{1}{2}} \leqslant U^R_{i+\frac{1}{2}}, \quad (b) \quad a^-_{i+\frac{1}{2}} \leqslant U^L_{i+\frac{1}{2}}, \quad (c) \quad a^+_{i+\frac{1}{2}} \geqslant U^R_{i+\frac{1}{2}}, \quad (d) \quad a^+_{i+\frac{1}{2}} \geqslant U^L_{i+\frac{1}{2}}.$$

Proof 3.1. *(a)* We start by distinguishing the different cases of positive or negative velocities. In the first case $0 \leqslant U^R_{i+\frac{1}{2}}$, and by definition $a^-_{i+\frac{1}{2}} \leqslant 0$, therefore the thesis follows. Instead when $U^R_{i+\frac{1}{2}} < 0$ it is sufficient to show that $\lambda_{\min}\left(\mathbf{f}'\left(\mathbf{Q}^R_{i+\frac{1}{2}}\right)\right) \leqslant U^R_{i+\frac{1}{2}}$.

A simple check shows that, for any $\mathbf{q}$, the matrix $\mathbf{f}'(\mathbf{q})$ is block lower triangular; if $\mathbf{f}'_u(\mathbf{q})$ denotes its leading (full) 2×2 principal submatrix, then the relations

$$\lambda_{\min}\left(\mathbf{f}'(\mathbf{q})\right) = \min\left\{\lambda_{\min}\left(\mathbf{f}'_u(\mathbf{q})\right), \beta_T U\right\}, \quad \lambda_{\max}\left(\mathbf{f}'(\mathbf{q})\right) = \max\left\{\lambda_{\max}\left(\mathbf{f}'_u(\mathbf{q})\right), \beta_T U\right\}$$

hold.

Now we suppose by contradiction that $\lambda_{\min}\left(\mathbf{f}'\left(\mathbf{Q}^R_{i+\frac{1}{2}}\right)\right) > U^R_{i+\frac{1}{2}}$, so that on one hand $\beta_T U^R_{i+\frac{1}{2}} > U^R_{i+\frac{1}{2}}$ which implies $\beta_T < 1$ because the velocity is assumed negative. On the other hand, at the same time we must have $\lambda_{\min}\left(\mathbf{f}'_u\left(\mathbf{Q}^R_{i+\frac{1}{2}}\right)\right) > U^R_{i+\frac{1}{2}}$ so that every eigenvalue of $\mathbf{f}'_u$ is greater than the velocity; this would imply

$$tr\left(\mathbf{f}'_u\left(\mathbf{Q}^R_{i+\frac{1}{2}}\right)\right) > 2\, U^R_{i+\frac{1}{2}},$$

where $tr(\cdot)$ denotes the trace of a matrix which equals the sum of its eigenvalues. A direct computation of the diagonal entries in the Jacobian shows that

$$tr\left(\mathbf{f}'_u\left(\mathbf{Q}^R_{i+\frac{1}{2}}\right)\right) = 2\beta_u U^R_{i+\frac{1}{2}}, \tag{3.15}$$

but since we are considering a negative velocity this is equivalent to $\beta_u < 1$, which conflicts with the lemma hypothesis since we have proved that $\beta_T < 1$.

(b) The proof is the same as the previous one, with the difference that $\lambda_{\min}\left(\mathbf{f}'\left(\mathbf{Q}^L_{i+\frac{1}{2}}\right)\right)$ must be compared with $U^L_{i+\frac{1}{2}}$.

(c) Again we distinguish the cases of positive and negative velocity. In the case where $U^R_{i+\frac{1}{2}} \leqslant 0$, the thesis immediately follows by the definition of the propagation speed, for which $a^+_{i+\frac{1}{2}} \geqslant 0$. Instead if $U^R_{i+\frac{1}{2}} > 0$, we apply the same argument as in *(a)*: suppose by contradiction that $\lambda_{\max}\left(\mathbf{f}'\left(\mathbf{Q}^R_{i+\frac{1}{2}}\right)\right) < U^R_{i+\frac{1}{2}}$ so that both the conditions $\beta_T U^R_{i+\frac{1}{2}} < U^R_{i+\frac{1}{2}}$ (from which $\beta_T < 1$ follows) and $\lambda_{\max}\left(\mathbf{f}'_u\left(\mathbf{Q}^R_{i+\frac{1}{2}}\right)\right) < U^R_{i+\frac{1}{2}}$ hold. On one hand,

$$tr\left(\mathbf{f}'_u\left(\mathbf{Q}^R_{i+\frac{1}{2}}\right)\right) < 2\, U^R_{i+\frac{1}{2}};$$

on the other hand, from (3.15) the positive sign of velocity yields $\beta_u < 1$, which contradicts the condition on the β coefficients.

(d) The proof is the same as the previous one, with the difference that we must compare $\lambda_{\max}\left(\mathbf{f}'\left(\mathbf{Q}^L_{i+\frac{1}{2}}\right)\right)$ with $U^L_{i+\frac{1}{2}}$.

□

Now consider the system of ODEs (3.4) and the central-upwind semi-discrete scheme described above. Assume, as a first instance, that the system is solved by the forward Euler method, which produces a time sequence $\{\mathbf{Q}^n_i\}_i$ of approximations (from which the current value of the height h^n_i is retrieved).

Theorem 3.2. *Suppose that, for all i, $h^n_i \geqslant 0$. Then, for all i, $h^{n+1}_i \geqslant 0$, provided that*

(i) $\max(\beta_u, \beta_T) \geqslant 1$,

(ii) $\Delta t \leqslant \dfrac{\Delta x}{2a}$, *where* $a := \max_i \left\{ \max\left\{ a^+_{i+\frac{1}{2}}, -a^-_{i+\frac{1}{2}} \right\} \right\}$ *and* $a^\pm_{i+\frac{1}{2}}$ *are the local speeds estimated at step 3.*

Proof 3.2. We report only a brief sketch of the proof because it is mainly similar to that presented in Kurganov and Petrova [158] under the same premises, where further details are offered.

Since we are interested in the behaviour of h^{n+1}_i, we focus on the first equation of the system discretized by the forward Euler method; then we rearrange the expression by using the definition (3.10) of the numerical flux at the interfaces and by considering the value of h^n_i as the average of the reconstructed values at the interfaces. We obtain the following expression for h^{n+1}_i, where $\lambda = \Delta t/\Delta x$

$$\begin{aligned} h^{n+1}_i = h^R_{i-\frac{1}{2}}\left[\frac{1}{2} + \left(\lambda a^-_{i-\frac{1}{2}}\right)\left(\frac{a^+_{i-\frac{1}{2}} - U^R_{i-\frac{1}{2}}}{a^+_{i-\frac{1}{2}} - a^-_{i-\frac{1}{2}}}\right)\right] + h^L_{i+\frac{1}{2}}\left[\frac{1}{2} - \left(\lambda a^+_{i+\frac{1}{2}}\right)\left(\frac{U^L_{i+\frac{1}{2}} - a^-_{i+\frac{1}{2}}}{a^+_{i+\frac{1}{2}} - a^-_{i+\frac{1}{2}}}\right)\right] \\ + h^L_{i-\frac{1}{2}}\left(\lambda a^+_{i-\frac{1}{2}}\right)\left(\frac{U^L_{i-\frac{1}{2}} - a^-_{i-\frac{1}{2}}}{a^+_{i-\frac{1}{2}} - a^-_{i-\frac{1}{2}}}\right) + h^R_{i+\frac{1}{2}}\left(-\lambda a^-_{i+\frac{1}{2}}\right)\left(\frac{a^+_{i+\frac{1}{2}} - U^R_{i+\frac{1}{2}}}{a^+_{i+\frac{1}{2}} - a^-_{i+\frac{1}{2}}}\right) \end{aligned} \tag{3.16}$$

where all the RHS terms are considered at time t_n, but, in order to simplify the notation, we have omitted the superscripts n. Thus h^{n+1}_i is a linear combination of the previous

values at the interfaces $h^{R/L}_{i\pm\frac{1}{2}}$ with suitable coefficients. The special reconstruction procedure at the interfaces guarantees that all $h^{R/L}_{i\pm\frac{1}{2}} \geqslant 0$. From the definition of the propagation speeds (3.9) it follows that $a^+_{i-\frac{1}{2}} \geqslant 0$, $a^-_{i+\frac{1}{2}} \leqslant 0$, $a^+_{i\pm\frac{1}{2}} - a^-_{i\pm\frac{1}{2}} \geqslant 0$. Thanks to assumption (i), we apply Lemma 3.1 which implies

$$0 \leqslant \frac{a^+_{i\pm\frac{1}{2}} - U^R_{i\pm\frac{1}{2}}}{a^+_{i\pm\frac{1}{2}} - a^-_{i\pm\frac{1}{2}}} \leqslant 1 \quad \text{and} \quad 0 \leqslant \frac{U^L_{i\pm\frac{1}{2}} - a^-_{i\pm\frac{1}{2}}}{a^+_{i\pm\frac{1}{2}} - a^-_{i\pm\frac{1}{2}}} \leqslant 1.$$

By adding the further CFL condition $\lambda a \leqslant 1/2$ ensured by assumption (ii), we conclude that all the coefficients in (3.16) are non-negative, therefore the fluid depth computed at the next time level h^{n+1}_i is a sum of non-negative terms for all i, and the proof is completed.

□

Remark 3.1. In the 2D case, assumption (ii) becomes $\Delta t \leqslant \min\left\{\frac{\Delta x}{4a}, \frac{\Delta y}{4b}\right\}$, where $\Delta x, \Delta y$ are the spatial steps along the two horizontal directions whereas a, b are the maximum local speeds of propagation at interfaces along the x- and y-axis respectively. Assumption (i) does not change and the proof of the analogous of Lemma 3.1 is essentially the same: the Jacobians are 4×4 matrices but keep a block triangular structure (up to permutation as $\mathbf{H}'(\mathbf{q})$ is concerned), according to the spectra displayed in (2.45).

Moreover, our derivation of β_u from a velocity profile never leads to a value less than 1, so that assumption (i) is always true. In fact, for any arbitrary profile $u(x, z, t)$ the equality

$$\int_B^{B+h} u^2(x, z, t)\, dz = \beta_u\, h\, U^2$$

must be imposed, as we have seen in (2.9) for the parabolic profile. If we consider for each (x, t) the L^2 space over the set $\{z : \ B \leqslant z \leqslant B + h\}$, the left side of the equation sees the squared L^2 norm of u; by Schwarz inequality,

$$\|u\|^2_{L^2} \geqslant \frac{|\langle u, 1\rangle_{L^2}|^2}{\|1\|^2_{L^2}}$$

where the inner product in the numerator corresponds to $\int_B^{B+h} u(x, z, t)dz = h\,U(x, t)$ whereas the denominator equals h. Therefore $\beta_u\, h\, U^2 \geqslant (hU)^2/h$ so that $\beta_u \geqslant 1$.

The proposal in literature by Costa and Macedonio [55] for lava flows considered all the correction factors in the range 0.5–1.5, independently on their derivation. A value of $\beta_u < 1$ makes the PDE system non-hyperbolic, but our theorem is still valid provided that such a choice is balanced by a parameter $\beta_T \geqslant 1$ in order to verify assumption (i), even though this is not the case for the particular temperature profile we have assumed.

Theorem 3.2 is stated assuming the forward Euler method for the time discretization of hyperbolic terms; instead, we use higher-order methods belonging to the family of Strong Stability Preserving (SSP) Runge-Kutta (R-K) schemes, see §3.2. However, a similar positivity preserving property is proved for all these methods, since any SSP R-K method is expressible as a convex combination of intermediate Forward Euler steps, see Gottlieb et al. [108]. Hence, Theorem 3.2 can be applied to each step of the combination

provided that Δt is sufficiently small (in order to satisfy the theorem assumption (ii) for each intermediate Euler method), ensuring the positivity preserving property for the whole SSP R-K method.

Remark 3.2. It is important to observe that, for each Euler term in the combination, the time step condition required by Theorem 3.2 depends on a, i.e., the maximum propagation velocity of the solution at interfaces evaluated at the considered term. Therefore the explicit determination of a global restriction for Δt (valid for the whole combination) is not so simple since we should estimate the evolution of a values during the intermediate steps of the R-K method.

3.2 Time discretization

In this section we present an Implicit-Explicit (IMEX) Runge-Kutta schemes to solve the system (3.4). These methods have been proposed by Pareschi and Russo [204] to solve stiff systems of differential equations written in this general form:

$$\frac{d}{dt}\mathbf{Q} = \mathcal{F}(\mathbf{Q}) + \mathbf{S}(\mathbf{Q}), \tag{3.17}$$

where $\mathbf{S}(\mathbf{Q})$ is the "stiff term". We write our problem in the same way by defining $\mathcal{F}(\mathbf{Q})$ as the spatial discretization of $-\partial_x \mathbf{f}(\mathbf{q}) + \mathbf{E}(\mathbf{q})$ in (3.2). In our situation the stiffness of $\mathbf{S}(\mathbf{Q})$ is due to the high viscosity, when $\gamma \gg 1$ in the momentum equation. In fact, the characteristic time of the source term is much smaller than the characteristic time of the advective part of the equations, hence the problem is said to be stiff and these kind of problems are generally difficult to solve numerically in an efficient way. A trivial solution is treating all the system explicitly and using a very small time step to ensure the stability of the scheme, but this approach results in long computational times.

In our scheme this limit has been overcome by applying an Implicit-Explicit method. An IMEX ν-stage Runge-Kutta scheme consists of applying an implicit discretization to the stiff term and an explicit one to the non stiff terms. Applying the scheme to the system (3.17) we obtain

$$\mathbf{Q}^{(l)} = \mathbf{Q}^n + \Delta t \sum_{k=1}^{l-1} \tilde{a}_{lk} \mathcal{F}(\mathbf{Q}^{(k)}) + \Delta t \sum_{k=1}^{\nu} a_{lk} \mathbf{S}(\mathbf{Q}^{(k)}), \quad l = 1, \dots, \nu \tag{3.18}$$

$$\mathbf{Q}^{n+1} = \mathbf{Q}^n + \Delta t \sum_{k=1}^{\nu} \tilde{\omega}_k \mathcal{F}(\mathbf{Q}^{(k)}) + \Delta t \sum_{k=1}^{\nu} \omega_k \mathbf{S}(\mathbf{Q}^{(k)}). \tag{3.19}$$

The choice of the $\nu \times \nu$ matrices $\tilde{A} = (\tilde{a}_{lk})$, where $\tilde{a}_{lk} = 0$ for $l \geqslant k$, and $A = (a_{lk})$ and of the vectors $\tilde{\omega} = (\tilde{\omega}_1, \dots, \tilde{\omega}_\nu)^T$ and $\omega = (\omega_1, \dots, \omega_\nu)^T$ characterizes the different schemes.

The IMEX Runge-Kutta schemes is presented by a *double tableau* in the usual Butcher notation,

$$\begin{array}{c|c} \tilde{c} & \tilde{A} \\ \hline & \tilde{\omega}^T \end{array} \qquad \begin{array}{c|c} c & A \\ \hline & \omega^T \end{array}$$

where the coefficients $\tilde{c}$ and c, used for the treatment of non autonomous systems (which is not our case), are given by the usual relations

$$\tilde{c}_l = \sum_{k=1}^{l-1} \tilde{a}_{lk}, \qquad c_l = \sum_{k=1}^{l} a_{lk}.$$

In all these schemes the implicit tableau corresponds to an L-stable scheme, that is $\omega^T A^{-1} e = 1$, where e is a vector whose components are all equal to 1. Tables 3.1 and 3.2 report examples of schemes, taken from Russo [238], adopted for the proposed numerical simulations in §3.4. Even though we choose couples with the same number of stages ν for both schemes according to Eqs. (3.18)–(3.19), it is possible to use schemes with different stages. For this reason in these Tables the classical notation (s, σ, p) is adopted, where s indicates the number of stages of the implicit scheme, σ the number of stages of the explicit scheme and p is the order of the IMEX scheme.

Table 3.1: Tableau for the Explicit (left) Implicit (right) IMEX-SSP(2,2,1) R-K scheme.

0	0	0
1	1	0
	1	0

0	0	0
1	0	1
	0	1

Table 3.2: Tableau for the Explicit (left) Implicit (right) IMEX-SSP(3,3,2) R-K scheme.

0	0	0	0
1/2	1/2	0	0
1	1/2	1/2	0
	1/3	1/3	1/3

1/4	1/4	0	0
1/4	0	1/4	0
1	1/3	1/3	1/3
	1/3	1/3	1/3

The algebraic equations obtained at each step (which must be solved implicitly) are treated simply and efficiently by adopting diagonally implicit Runge-Kutta (DIRK) schemes ($a_{lk} = 0$, for $k > l$), which lead to a sequence of equations for a single $\mathbf{Q}^{(l)}$ and, in addition, guarantee that $\mathcal{F}$ is always evaluated explicitly.

3.2.1 IMEX Runge-Kutta schemes

When using DIRK schemes, at each internal Runge-Kutta step we have an implicit problem for all the cells of the domain. In fact, by applying a DIRK scheme to the ordinary differential equations (3.4) we obtain

$$\mathbf{Q}_i^{(l)} = \mathbf{Q}_i^n + \Delta t \sum_{k=1}^{l-1} \tilde{a}_{lk} \left(- \frac{\mathbf{F}_{i+\frac{1}{2}}^{(k)} - \mathbf{F}_{i-\frac{1}{2}}^{(k)}}{\Delta x} + \mathbf{E}_i^{(k)} \right) + \Delta t \sum_{k=1}^{l} a_{lk} \mathbf{S}(\mathbf{Q}_i^{(k)}),\ l = 1, \dots, \nu \quad (3.20)$$

$$\mathbf{Q}_i^{n+1} = \mathbf{Q}_i^n + \Delta t \sum_{k=1}^{\nu} \tilde{\omega}_k \left(- \frac{\mathbf{F}_{i+\frac{1}{2}}^{(k)} - \mathbf{F}_{i-\frac{1}{2}}^{(k)}}{\Delta x} + \mathbf{E}_i^{(k)} \right) + \Delta t \sum_{k=1}^{\nu} \omega_k \mathbf{S}(\mathbf{Q}_i^{(k)}). \quad (3.21)$$

Furthermore $\mathbf{S}(\mathbf{Q}_i^{(k)})$ depends only on the solution in the cell i, thus, for each Runge-Kutta step, we have to solve n independent implicit problems, one for each cell of the domain.

All the implicit problems are solved by using a functional iterative method. For simplicity, we rewrite the equation (3.20) as follows:

$$\mathbf{Q}_i^{(l)} = \mathbf{Q}_i^n + \Delta t \sum_{k=1}^{l-1} \tilde{a}_{lk} \left(- \frac{\mathbf{F}_{i+\frac{1}{2}}^{(k)} - \mathbf{F}_{i-\frac{1}{2}}^{(k)}}{\Delta x} + \mathbf{E}_i^{(k)} \right) + \Delta t \sum_{k=1}^{l-1} a_{lk} \mathbf{S}(\mathbf{Q}_i^{(k)}) + \Delta t \cdot a_{ll} \mathbf{S}(\mathbf{Q}_i^{(l)}),$$

which is simply expressible as

$$\mathbf{Q}_i^{(l)} - \Lambda_i - \Delta t \cdot a_{ll} \mathbf{S}(\mathbf{Q}_i^{(l)}) = 0 \quad \Longleftrightarrow \quad \Gamma(\mathbf{Q}_i^{(l)}) = 0$$

where the term Λ_i contains the explicit part of the equation (3.20), which can be evaluated once during the iterative computation of the solution $\mathbf{Q}_i^{(l)}$.

We apply the Newton-Raphson method [200] to the problem $\Gamma(\mathbf{x}) = 0$ obtaining

$$\begin{cases} \mathbf{x}_{(0)} = \mathbf{Q}_i^n \\ \mathbf{x}_{(k+1)} = \mathbf{x}_{(k)} - J^{-1}_{|_{\mathbf{x}_{(k)}}} \cdot \Gamma(\mathbf{x}_{(k)}) \end{cases} \tag{3.22}$$

where $J^{-1}_{|_{\mathbf{x}_{(k)}}}$ is the inverse of the Jacobian of the function Γ evaluated at $\mathbf{x}_{(k)}$. The Newton-Raphson method is second order convergent, its computational cost depends on the cost of computing derivatives in the entries of the Jacobian and on the cost of the Jacobian inversion, in addition Newton-Raphson method suffers for a bad choice of the initial guess for the iterations (this motivates our choice given by the variables computed at the previous time). We underline here that the inversion of the Jacobian is not computationally expensive, since J is a $m \times m$ matrix, where m is the number of the equations having terms treated implicitly, for example in the 2D case $m = 3$.

3.3 Derivative approximation through complex numbers

The solution of the linear system appearing in (3.22) requires an accurate numerical evaluation of the Jacobian matrix J, i.e. the partial derivatives of Γ with respect to the components of $\mathbf{x}$. Due to the strong non-linearity of $\mathbf{S}(\mathbf{Q})$, it is not convenient to determine analytically the derivatives.

A common method to estimate the first derivative is the first-order forward-difference formula and higher order extensions which increase the stencil by using Taylor series expansion. However, when estimating sensitivities using finite-difference formulas we are faced with the "step-size dilemma", that is the desire to choose a small step size to minimize truncation error while avoiding the use of a step so small that errors due to subtracted cancellation become dominant.

A way to overcome this problem is to use complex functions. The first use of complex variables to estimate derivatives starts with the work of Lyness [180], Lyness and Moler [181]. They introduced a reliable method for computing the derivatives of an analytic function, and later Squire and Trapp [248] obtained a very simple expression for estimating

the first derivative. It has been shown that this estimation is very accurate, extremely robust and easy to implement, with a reasonable computational cost. Recently it has been used for sensitivity analysis in computational fluid dynamics and further research on the subject has been carried out [3, 185, 186].

Let's see more in detail the theory behind the estimation of derivatives using complex variables. Consider a function $g(x) : \mathbb{C} \to \mathbb{C}$, $g \in C^1(\mathbb{C})$ and assume that $u(x+iy) = \mathcal{R}(g(x+iy))$ and $v(x+iy) = \mathcal{I}(g(x+iy))$ so that $g(z) = u(x+iy) + iv(x+iy)$, where i denotes the imaginary unit and $\mathcal{R}(\cdot)$, $\mathcal{I}(\cdot)$ stand for the real or imaginary part of a complex number respectively.

Then the Cauchy-Riemann equalities hold:

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y}, \qquad \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x}; \tag{3.23}$$

thanks to these equalities we can write for instance

$$\left.\frac{\partial u}{\partial x}\right|_{(x+iy)} = \lim_{h\to 0} \frac{v(x+i(y+h)) - v(x+iy)}{h}, \tag{3.24}$$

where h is a real number.

When a function $f \in C^1(\mathbb{R})$ can be extended to a new $\tilde{f} : \mathbb{C} \to \mathbb{C}$, $\tilde{f} \in C^1(\mathbb{C})$ such that $\tilde{f}(x) = f(x)$, $\forall x \in \mathbb{R}$, by using the relation (3.24) we have

$$\left.\frac{\partial \mathcal{R}(\tilde{f})}{\partial x}\right|_{(x+iy)} = \lim_{h\to 0} \frac{\mathcal{I}(\tilde{f}(x+i(y+h))) - \mathcal{I}(\tilde{f}(x+iy))}{h} \tag{3.25}$$

and finally by posing $y = 0$ we obtain

$$\begin{aligned} \left.\frac{\partial \mathcal{R}(\tilde{f})}{\partial x}\right|_{(x)} &= \lim_{h\to 0} \frac{\mathcal{I}(\tilde{f}(x+ih)) - \mathcal{I}(\tilde{f}(x))}{h} \\ \implies \left.\frac{\partial f}{\partial x}\right|_{(x)} &= \lim_{h\to 0} \frac{\mathcal{I}(\tilde{f}(x+ih))}{h}. \end{aligned}$$

For a small discrete h, this can be approximated by

$$\left.\frac{\partial f}{\partial x}\right|_{(x)} \approx (f'(x))_{CS} = \frac{\mathcal{I}(\tilde{f}(x+ih))}{h}.$$

This is called the *complex-step derivative approximation*. This estimation is not subject to subtractive cancellation errors, since it does not involve a difference operation. To see the improvements with respect to the finite differences, we try to approximate the first derivative of the analytic function

$$f(x) = \frac{x^2}{1+x^4}$$

at $x = 0.25$. We compare the complex-step derivative approximation with the first order forward differences

$$(f'(x))_{F_1} = \frac{f(x+h) - f(x)}{h},$$

the first order backward differences

$$(f'(x))_{B_1} = \frac{f(x) - f(x-h)}{h},$$

the second order central differences

$$(f'(x))_{C_2} = \frac{f\left(x + \frac{h}{2}\right) - f\left(x - \frac{h}{2}\right)}{h},$$

the second order forward differences

$$(f'(x))_{F_2} = \frac{-f(x+2h) + 4f(x+h) - 3f(x)}{2h},$$

the second order backward differences

$$(f'(x))_{B_2} = \frac{3f(x) - 4f(x-h) + f(x-2h)}{2h},$$

and finally the fourth order central differences

$$(f'(x))_{C_4} = \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h}.$$

In Figure 3.5 we report the normalized relative error

$$\epsilon = \left| \frac{(f'(x))_{est} - f'(x)}{f'(x)} \right|$$

of all the estimations presented above with respect to the exact value $f'(x)$ calculated at $x = 0.25$. As we can see, at the beginning the relative error of all the estimations decrease with the step-size, but at a certain point, for the finite differences, the subtractive cancellation errors become significant, and thus the relative error increases. With the complex step derivative approximation, instead, this does not happen, and the relative error continues to decrease until it reaches the machine working precision. Then, even decreasing the step size, the error remains almost constant. Thus, with the complex step derivative approximation we do not have anymore the "step-size dilemma" and we can choose a step-size close to the machine working precision obtaining the highest accuracy for the approximation of the first derivative.

Through the complex-step derivative approximation we compute the Jacobian J, needed for the Newton-Raphson method, with an error of the same order of the machine working precision. All we have to do is to extend the function Γ to the complex plane, by introducing the new function $\tilde{\Gamma} : \mathbb{C} \to \mathbb{C}$ and to compute each column of the Jacobian at $\mathbf{x}$ as

$$J(\mathbf{x}) \cdot \mathbf{e}_j \approx \frac{\mathcal{I}(\tilde{\Gamma}(\mathbf{x} + i\, h\, \mathbf{e}_j))}{h} \tag{3.26}$$

where $(\mathbf{e}_j)_{j=1,\dots,m}$ are the canonical basis vectors.

3.4 Numerical simulations

The Fortran 90 numerical code developed is a variant of the solver IMEX_SfloW2D and it has been tested and validated with some literature examples and other tests to verify: (i) the well-balancing and the positivity preserving properties of the scheme; (ii) the

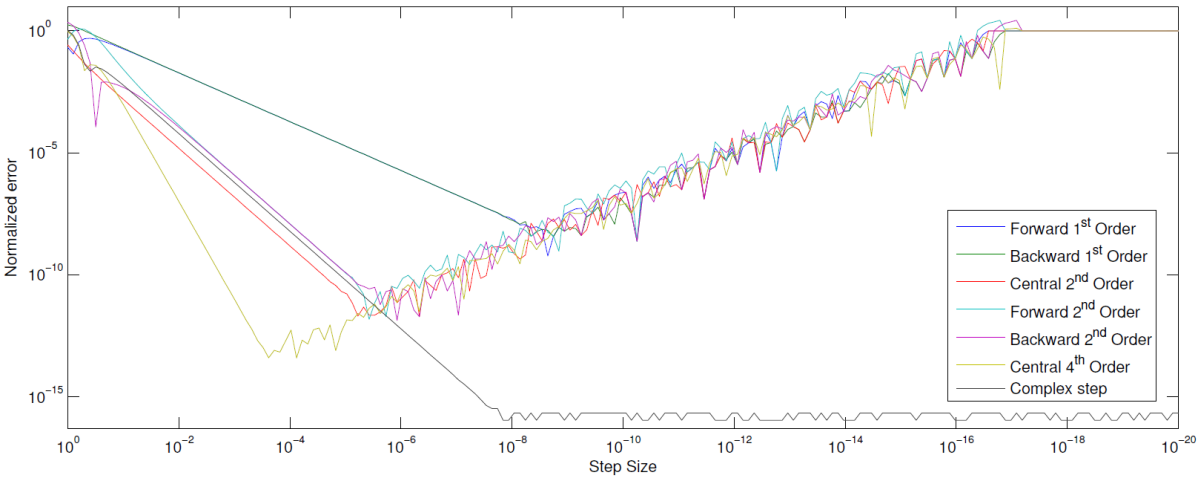


Figure 3.5: Normalized relative error of the derivative approximations with respect the exact value.

consequences of using different limiters; (iii) the various behaviours when adopting constant vertical profiles of the variables or not. One recalls that in our scheme the linear reconstruction at interfaces is computed over the set of variables $\mathbf{P}$ (see §3.1) whereas in the scheme proposed by Kurganov and Petrova [158] it is computed over the variables $[w, hU, hV]$, hence we will stress over differences or similarities of such different computations; in the following, we indicate them with PVR and CVR respectively. Furthermore, we underline that our code adopts a uniform grid for spatial discretization.

When the outputs of simulations done with different parameters or methods are compared, a metric is needed to compute the difference between two numerical solutions. Here, if $\mathbf{y}^A$ and $\mathbf{y}^B$ are vectors containing solution values on the same grid, the following expression is used

$$diff(\mathbf{y}^A, \mathbf{y}^B) := \left[\Delta x \sum_i \left(\frac{|y_i^A - y_i^B|}{\max(|y_i^A|, |y_i^B|)} \right)^p \right]^{1/p} \tag{3.27}$$

with $p = 2$. This expression is a combination of the p-norm, recalled also by LeVeque [168], and the relative distance proposed by Ziv [284].

The numerical tests are presented increasing complexity ordered so that a more accurate evaluation of the single performances of each implemented feature can be achieved. Also, those tests that we address as "BM*" originate from the paper Cordonnier et al. [52] that proposes benchmark tests for lava-flow models. The first test, in §3.4.1, is a preliminary test of a Riemann problem that is necessary to check the wave propagation and a good treatment of discontinuities, both in solution and bottom. The second and third tests, in §3.4.2 and §3.4.3 respectively, are 1D dam-break simulations of viscous fluids over flat and inclined plane. We have doubled the experiments to present simulations both with high and low viscosity, in order to check the sensitivity of the scheme to the values of β_u, namely the sensitivity to constant or parabolic velocity profile. We remark that we use the terms high viscosity and low viscosity to distinguish between the two cases, but in all the examples the values of viscosity are large enough to result in laminar flows. In fact, the fluid simulated in the examples are representative of lava, silicone gum and silicone oil (both used in analog experiments as benchmarks for models of lava flow emplacement). Moreover, the test case in §3.4.2 sees also a temperature-dependent density simulation.

These first three tests presented so far are 1D (while the remaining four are 2D), moreover, their spatial domains was considered closed by fixed walls, hence homogeneous Dirichlet conditions have been imposed for the velocity (in this case zero velocity in correspondence to the walls) and homogeneous Neumann conditions for $h+B$ (zero gradient). The fourth test, §3.4.4, consists of the spreading of a viscous and isothermal fluid over an inclined plane and its aim is to check the fluid spreading correctness with respect to three directions. The fifth test, see §3.4.5 is a 2D simulation of a hot viscous fluid, and its focus is on the effects of a temperature-dependent viscosity. The sixth test, in §3.4.6, sees the spreading of a hot viscous fluid over a flat plane that cools during the axisymmetric propagation because of radiative, convective and conductive heat loss. We also underline that viscosity is not temperature-dependent hence the dynamics are not influenced by that. The aims of the test are to check both the fluid propagation and temperature changes. The seventh and last test §3.4.7 is a 2D simulation of a real lava flow in the context of a realistic effusive eruption, hence we modeled a Bingham plastic hot fluid and accounted for the thermal heat exchanges with the environment and soil, the temperature-dependent viscosity, and the viscous heating. We adopted the Fogo topography and data from the 2014–2015 eruption to test the impact of the rheological parameters on the simulations. In the last four test cases, we have modeled an open domain by imposing zero-gradient Neumann conditions both for velocities, $h + B$ and temperature.

In order to preserve the positivity of the solution and the stability of the explicit scheme, associated with the CFL condition, a variable time step is used, which has been set to the following values:

$$\Delta t = \begin{cases} k\frac{\Delta x}{a} & \text{for the 1D simulations, with } k = 0.45, \\ k \min\left\{\frac{\Delta x}{a}, \frac{\Delta y}{b}\right\} & \text{for the 2D simulations, with } k = 0.24, \end{cases} \tag{3.28}$$

where we recall that a, b are the maximum local speeds of propagation at interfaces (see §3.1.2). In all tests, the initial time-step is set as $\Delta t = 10^{-4}$ s.

3.4.1 Riemann problem with discontinuous bottom

In this first test for an inviscid fluid, we want to check the correct treatment of discontinuities both in the initial state and in the topography, compare the PVR and CVR and the limiters effects. This is a very common test (for example see [188]), with a step function for the topography B and a discontinuous initial state

$$B(x) = \begin{cases} 0, & x \leqslant 6, \\ 1, & x > 6, \end{cases} \qquad h(x,0) = \begin{cases} 5, & x \leqslant 6 \\ 1, & x > 6 \end{cases}, \quad \text{and} \quad u(x,0) = 0, \tag{3.29}$$

whereas the gravitational constant is $g \approx 9.81$, as usual. The exact solution of this initial value problem consists of a left-going rarefaction wave followed by a stationary wave in correspondence to the bottom step and then by a right-going shock. Since we are simulating an inviscid fluid, we have to consider $\beta_u = 1$, namely a uniform velocity along the vertical direction. We show solutions computed over the horizontal interval $[0, 12]$ with different grid sizes, $\Delta x = 0.08$, $\Delta x = 0.02$ and $\Delta x = 0.0003125$, which correspond respectively to 150, 600 and 38400 cells. For the computation of the interface values in the formula (3.5) we have used both the generalized minmod limiter and also a piecewise constant reconstruction (namely no-limiter). We recall that the generalized minmod limiter is given by Eq. (3.6), where we have chosen the parameter value $\theta = 1.3$.

For the temporal discretization we have used the SSP R-K scheme with 3 stages reported in Table 3.2.

In Figure 3.6 the solutions at time $t = 0.5\,\mathrm{s}$ are obtained by the interface reconstructions computed through the $\mathbf{P}$ set of variables (PVR), on the opposite for Figure 3.7 we have done it through the conservative variables (CVR) and the difference between these solutions computed with (3.27), has magnitude 10^{-3}. When comparing solutions obtained with and without limiter, it is clear that the ones computed with the use of the *generalized minmod* limiter have a remarkably high accuracy achieved even for low-resolution simulations, in particular the solution computed with 150 cells and the *generalized minmod* limiter is better than the one obtained with 600 cells and no limiter.

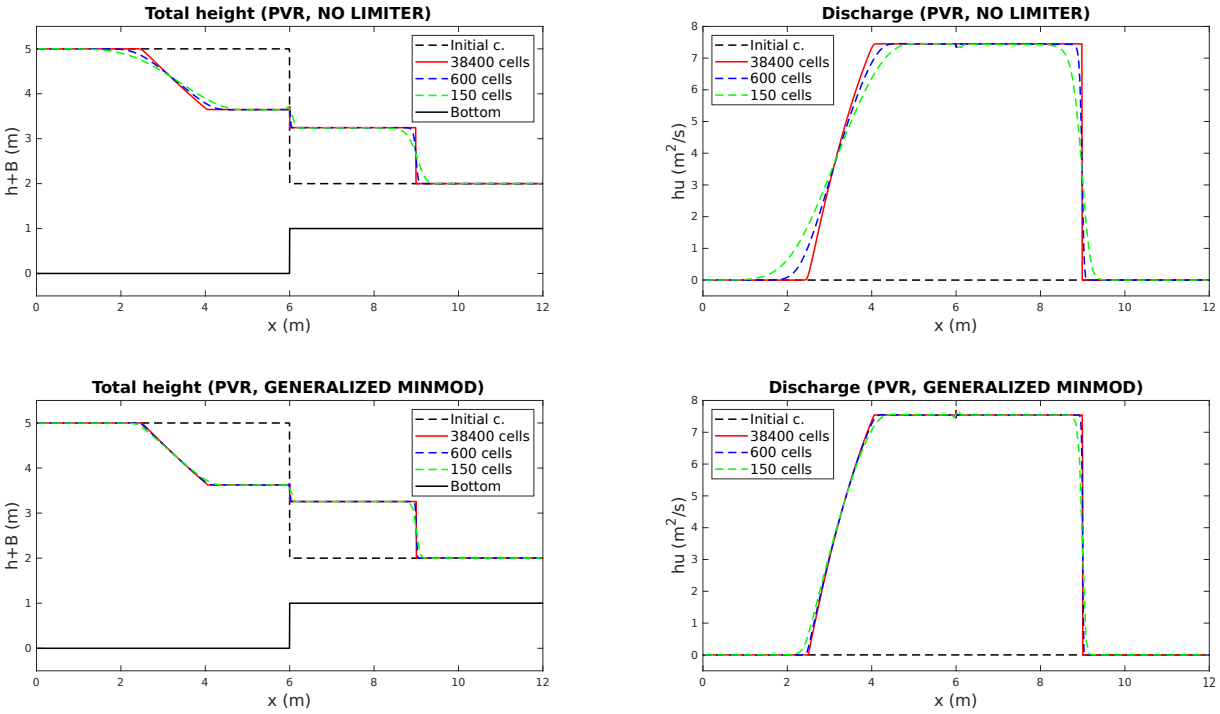


Figure 3.6: *Riemann problem with discontinuous bottom.* Water surface $h + B$, on the left, and discharge hu at time $t = 0.5\,\mathrm{s}$. Simulations computed by different grid size and by PVR at interfaces. Top: No limiter adopted, i.e. constant piecewise reconstruction. Down: linear reconstruction at interfaces done by the generalized minmod limiter.

For the simulation on 600 cells with PVR and with the generalized minmod limiter, we needed 473 time steps with a total execution time of 2.26 s, whereas the time-step size has rapidly reached the value of $1.11 \cdot 10^{-3}\,\mathrm{s}$. See Table 3.6 on page 158 that reports this information and that of the other tests.

3.4.2 BM1: Dam-break of viscous fluids over a flat bottom

We start from the simulations of viscous fluids with constant density. We explore the results focusing on some aims: (*i*) check the correct treatment of wet/dry states and of low/high viscosity; (*ii*) compare results (by computing the relative difference through Eq.(3.27)) obtained with PVR and CVR, 2 and 3 stages Runge-Kutta schemes reported in Tables 3.1-3.2 and constant and parabolic velocity profile; (*iii*) study the limiters effects. We conclude this section with the simulations of a temperature-dependent variable density fluid.

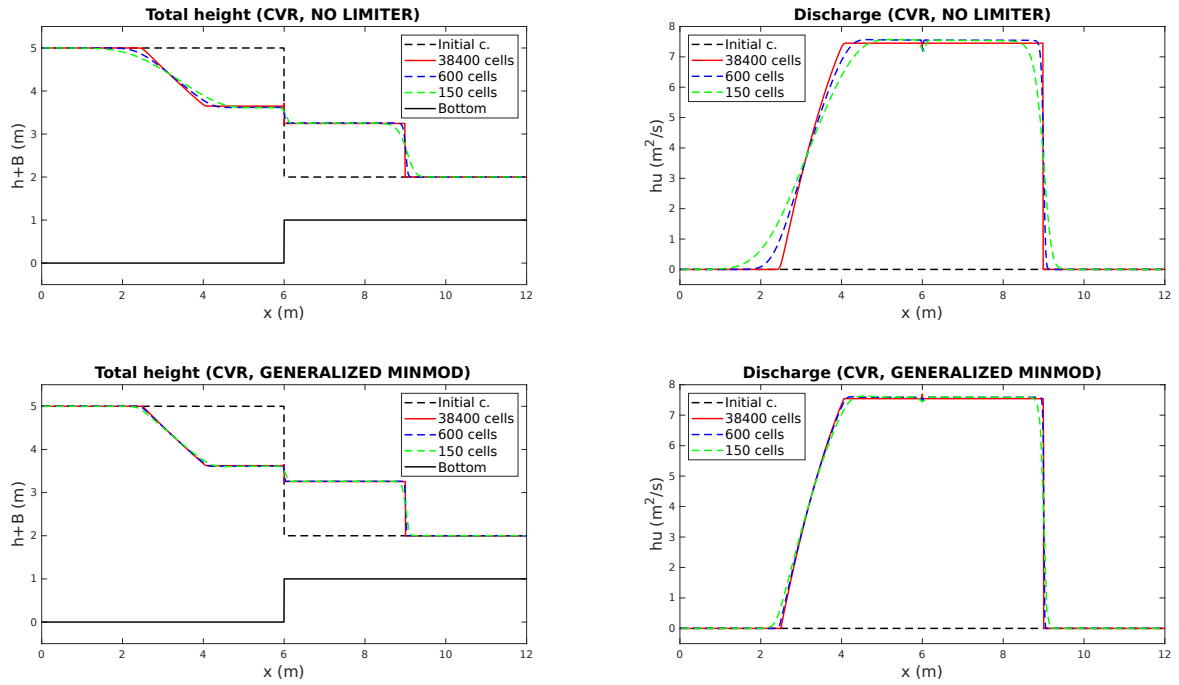


Figure 3.7: *Riemann problem with discontinuous bottom.* Water surface $h + B$, on the left, and discharge hu at time $t = 0.5\,\mathrm{s}$. Simulations computed by different grid size and by CVR at interfaces. Top: No limiter adopted, i.e. constant piecewise reconstruction. Down: linear reconstruction at interfaces done by the generalized minmod limiter.

The dam-break test simulates the rupture of a dam: the viscous fluid is initially confined in a box and then, after the abrupt removal of one box sides, it starts to flow into a channel. This is a classical fluid mechanics problem, widely used because of its mathematical tractability and numerous applications, from hydraulic engineering, to food science, from geophysical science, to industrial problems [6]. This test is also considered as a standard measurement when characterizing fluid rheology, and it is a common benchmark for the shallow-water numerical codes too [240].

For our simulations, we consider a viscous fluid with a Newtonian rheology (§1.1.5.3) and a dam $L = 6.6\,\mathrm{m}$ long and $h = 1\,\mathrm{m}$ high over a horizontal domain of length $L_{TOT} = 75\,\mathrm{m}$, as suggested in [52] where the dam-break test is recommended as an isothermal benchmark for lava-flow numerical codes. When the flow is slow enough to neglect inertia, the evolution of the front can be associated with the scaling law proposed by Saramito et al. [240] that describe the front evolution $x_f(t)$ by a power-law with exponent 0.5, for short times, and 0.2 for long times:

$$\frac{x_f(t)}{L} \approx \begin{cases} 0.284\left(\dfrac{t}{t_c}\right)^{1/2} & \text{if} \quad t < 2.5t_c, \\ 1.133\left(\dfrac{t}{t_c} + 1.221\right)^{1/5} - 1 & \text{otherwise,} \end{cases} \tag{3.30}$$

with the characteristic time t_c depending both on geometry of the initial condition and on fluid rheology: $t_c = \left(\dfrac{L}{h}\right)^2 \dfrac{\mu/\rho}{gh}$. In particular, the fluid collapse that occurs immediately after the dam rupture influences the short-term flow regime, and the fluid takes more time in propagating in this regime with the increase in viscosity. On the opposite, this sort of dynamics is almost instantaneous for a low viscosity fluid, and the gravity waves affect

the propagation so that the equation (3.30) is no more able to describe such a case.

We perform this test both with high and low viscosity, classifying the regime through the Froude number Fr defined by Eq. (2.47). For this application, we compute Fr using the front velocity of propagation and the mean thickness over the flow extent, as the front thickness is comparable with the mean value.

High viscosity For the high-viscosity test, we model a silicone gum, following the laboratory dam-break experiment presented by Saramito et al. [240] where a transparent synthetic polymer SGM 36 manufactured by Dow Corning (USA) was employed. In this test temperature is not considered and we have used a constant kinematic viscosity value $\nu = \mu/\rho = 3.7\,\mathrm{m}^2/\mathrm{s}$ (as suggested in Example 1, Cordonnier et al. [52]).

The relative differences after $t = 100\,\mathrm{s}$, anyway the velocity profile is chosen, between solutions computed by the PVR or CVR are very small ($\sim 10^{-4}$) and the same happens if we use R-K schemes with 2 or 3 stages ($\sim 10^{-7}$). In addition there are no appreciable differences in the solutions when using $\beta_u = 1$ or $\beta_u = 1.2$, i.e. constant or parabolic velocity profile, because the flow is in a strict subcritical regime with $Fr \approx 0.018$. This condition is mainly due to the high viscosity which, by opposing to the motion, keeps the flow velocity very low. Because of this, our results are comparable with the analytic solution of Eq. (3.30), see below.

In order to see the convergence behaviour, we compare results obtained with 200, 400, 800 and 3600 cells (using a 3-stages R-K, $\beta_u = 1$ and by PVR at interfaces) in Figure 3.8. We distinguish three different cases of computation of the interface values (see Eq. (3.5)): we have used both the *minmod* limiter, the *generalized minmod* limiter and also a piecewise constant reconstruction (namely no-limiter). By using the generalized minmod with $\theta = 2$ a very accurate solution is obtained with 400 cells whereas, with the minmod limiter, similar results are obtained with 800 cells and, in the case of no limiter, with 3200 cells. On the opposite, by using the generalized minmod limiter a less refined grid is sufficient to obtain a very good solution.

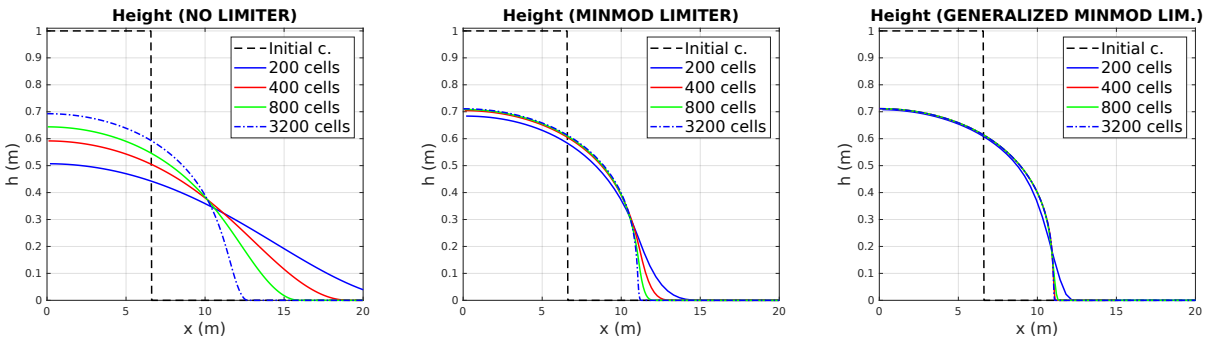


Figure 3.8: *Dam-break of a viscous fluid over a flat bottom.* Solution at $t = 100\,\mathrm{s}$ of a high-viscosity fluid. Solutions computed by different grid size over a domain 75 m long. *Left*: no limiter. *Center*: minmod limiter. *Right*: generalized minmod limiter.

Figure 3.9 shows the convergence of the front position to the scaling law of Eq. (3.30). Three different grid resolutions have been used: 400, 800, and 1600 cells, and the generalized minmod limiter has been adopted. The front position of the numerical solutions is determined by a threshold value $h = 10^{-3}$ m.

For the 400 cells grid, the simulation with PVR, generalized minmod limiter and $k = 0.45$ for the time step condition (3.28), took 3432 time steps with a total execution

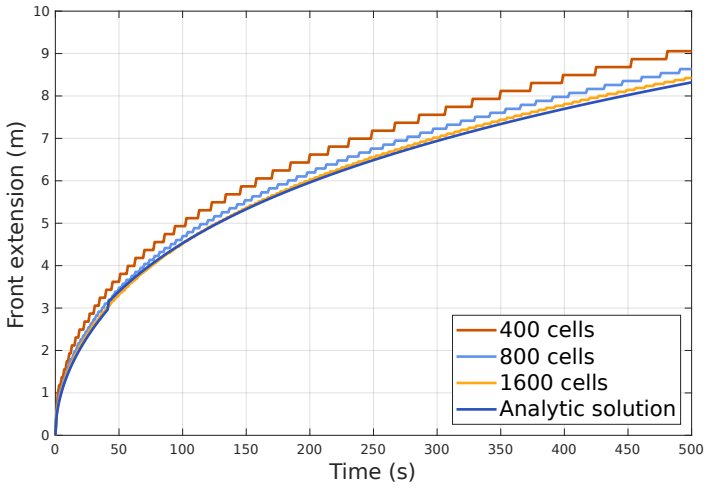


Figure 3.9: *Dam-break of a viscous fluid over a flat bottom.* Front position over time of a high-viscosity fluid: scaling law given by Eq. (3.30) and simulation results obtained with different grid size.

time of 7.91 s, whereas the time-step size grew slowly up to $3.2 \cdot 10^{-2}$ s. See Table 3.6 on page 158 that reports this information and that of the other tests.

Low viscosity For the low-viscosity test, we model a silicone oil, the material used by Lister [172] in his laboratory experiments for the spreading of isothermal viscous fluids on a plane. This silicone oil is characterized by a kinematic viscosity $\nu = \mu/\rho = 1.16 \cdot 10^{-3}\,\mathrm{m}^2/\mathrm{s}$ (taken from Example 2, Cordonnier et al. [52]). Also for this test, temperature is not considered.

One observes a low sensitivity to most numerical parameters. Regardless of the assumption about the velocity profile, the relative differences between solutions obtained through R-K schemes with 2 or 3 stages and PVR or CVR are about 10^{-3}. As we will see, the results are also independent from the limiter adopted. On reverse, the assumption of constant or parabolic velocity profiles leads to different dynamics, as shown in Figure 3.10. The main outcome of the two simulations is that solutions with different vertical profiles have distinct shapes and temporal evolution, indeed initially the flow front of the simulation with $\beta_u = 1.2$ (parabolic profile) proceeds faster, after 12 s the solutions are equal merit and then they reverse positions.

The sensitivity to the velocity vertical profile emerges because, in this case, the flow regime is supercritical, therefore the inertial forces control the flow dynamics and the influence of the β_u coefficient becomes notable, as seen in the discussion at the end of §2.1.6. In this case we are not aware of any analytical solution or scaling law for comparison and Eq. (3.30) is not applicable.

Figure 3.11, representing the front position and the Froude number, shows that differences in the Froude numbers between the two simulations result in diverse velocities of propagation of the flow front.

With respect to the high-viscosity case, the low-viscosity test highlights a lower sensitivity of the numerical solution to the choice of the limiter, as shown by Figure 3.12. In addition, there is no necessity to refine too much the discretized domain since the solutions computed with only 400 cells are all close to the convergence solution.

Using a 400 cells grid for the simulation with PVR, with the generalized minmod limiter and with $k = 0.45$ for the time step condition (3.28), in the case of $\beta_u = 1$

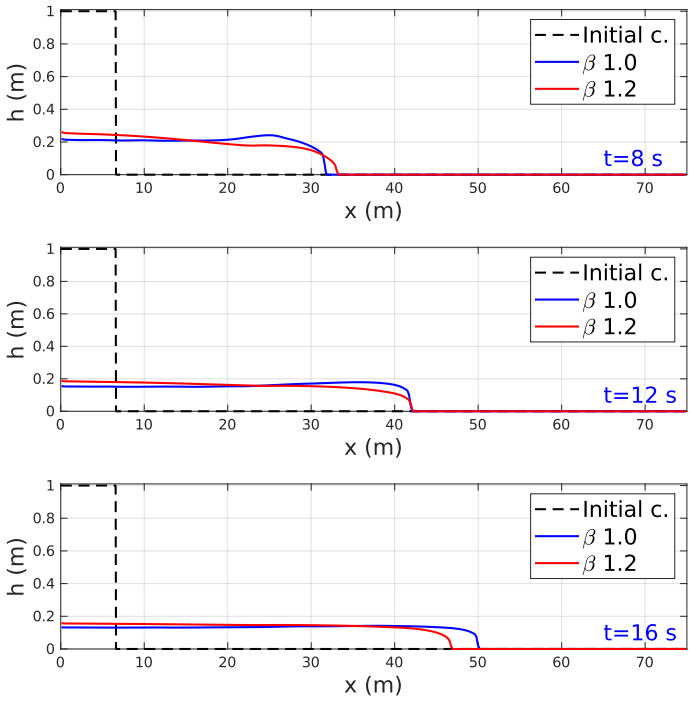


Figure 3.10: *Dam-break of a viscous fluid over a flat bottom.* Evolution: *top* $t = 8\,\mathrm{s}$, *center* $t = 12\,\mathrm{s}$, *bottom* $t = 16\,\mathrm{s}$ of a low-viscosity fluid, computed by PVR at interfaces with the minmod limiter, 3 stages IMEX R-K scheme. Comparison between $\beta_u = 1.0$ and $\beta_u = 1.2$ (constant and parabolic velocity profile).

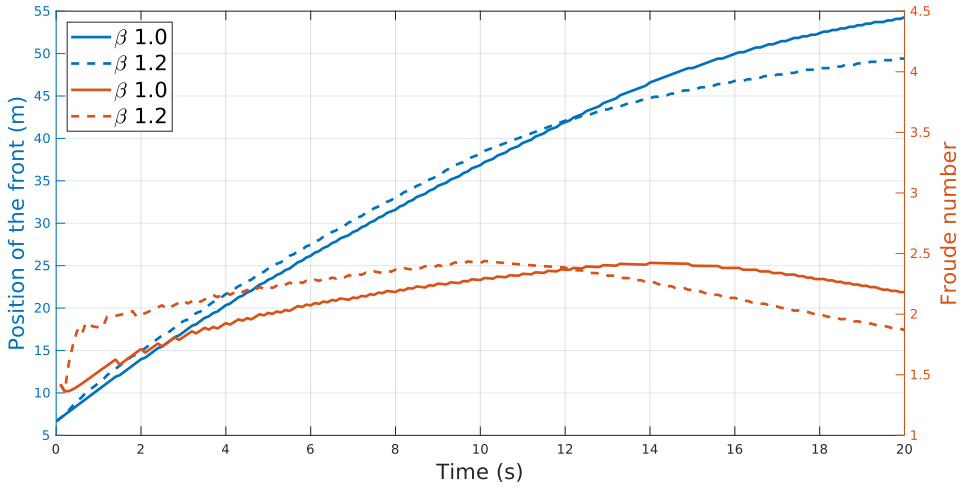


Figure 3.11: *Dam-break of a viscous fluid over a flat bottom.* Evolution of the front position (blue) and the Froude number (red) of a low-viscosity fluid. Comparison between constant velocity profile with $\beta_u = 1.0$ (solid line) and parabolic velocity profile $\beta_u = 1.2$ (dashed line).

(constant velocity profile) we needed 597 time steps with a total execution time of 1.9 s whereas the time-step size grew slowly up to $2.1 \cdot 10^{-2}$ s; in the case of $\beta_u = 1.2$ (parabolic velocity profile), we needed 757 time steps with a total execution time of 2.44 s whereas the time-step size grew slowly up to $1.9 \cdot 10^{-2}$ s.

Low viscosity with temperature dependent density Silicon oil, the material employed in the previous low-viscosity test, is characterized by a linear dependence of density on temperature in the range of $300 - 380\,\mathrm{K}$, as supported from Lecuna et al. [163]. That

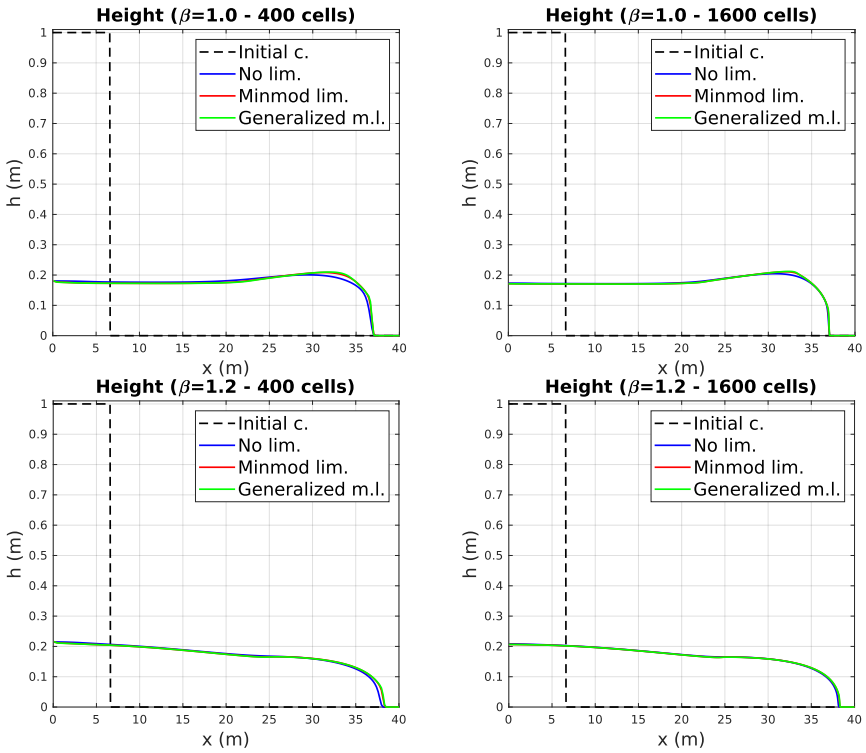


Figure 3.12: *Dam-break of a viscous fluid over a flat bottom.* Solutions at $t = 10\,\mathrm{s}$ of a low-viscosity fluid computed without limiter, with the minmod and the generalized minmod limiter. *Top:* $\beta_u = 1.0$. *Bottom:* $\beta_u = 1.2$.

work presented some experimental data about mineral oil and silicone oil and we have extrapolated from it the values for the linear relation between density and temperature, Eq. (2.41), $\rho(T) = 1200 - 0.75T$. We set the maximum temperature on the surface $T_1 = 338\,\mathrm{K}$ and the initial depth-averaged temperature $T = 330\,\mathrm{K}$ with the thermal boundary layer thickness $\delta_T = h/4$. In Figure 3.13 we present the results of the simulation at times $t = 8\,\mathrm{s}$ and $20\,\mathrm{s}$ computed with 400 cells by PVR at interfaces with the minmod limiter, 3 stages IMEX R-K scheme, $\beta_u = 1.2$ (parabolic velocity profile). Since we consider a laminar flow, the co-presence of velocity profile and thermal profile implies that the top layer with the highest temperature moves faster than the lower part and it rapidly constitutes a large portion of the front. For this reason the front has the highest temperature and consequently the lowest density; conversely, the tail average temperature decreases and then density grows. See Table 3.6 on page 158 that reports this information and that of the other tests.

The relative difference between the variable-density and the constant-density solutions increases with time reaching the maximum value $\sim 10^{-1}$ at $t = 10\mathrm{s}$. The simulation results highlight that, in this circumstance, density variations do not affect significantly nor the front position neither the global shape of the flow, therefore in the following tests we neglect density variations.

Laboratory test of a low viscosity fluid in supercritical regime In the absence of an analytic solution or a scaling law to compare the low viscosity dam-break case results, we decided to validate our model with a laboratory test. Our primary interest is the confirmation that the adoption of a parabolic profile ($\beta_u = 1.2$) produces better results in the case of supercritical regimes, that is, the situation when the simulations present the

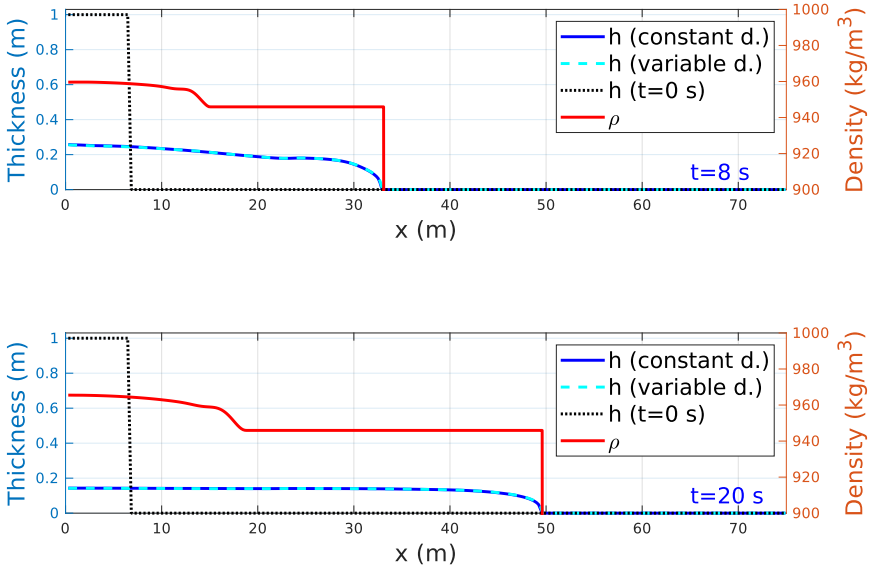


Figure 3.13: *Dam-break of a viscous fluid over a flat bottom.* Thickness and density for solutions of a low-viscosity fluid with temperature-dependent variable density and constant density. Simulation computed over 400 cells by PVR at interfaces with the minmod limiter, 3 stages IMEX R-K scheme, $\beta_u = 1.2$ (parabolic velocity profile). *Top:* $t = 8\,\mathrm{s}$. *Bottom:* $t = 20\,\mathrm{s}$.

most prominent differences (see Figure 3.12).

The laboratory test was done during the visit at the Lamont-Doherty Earth Observatory (at the Columbia University, New York), with the supervision of Prof. Einat Lev and the assistance of Janine Birnbaum, and has been repeated twice. We can present only preliminary results because the worldwide pandemic emergency prevented us from pursuing the experiments.

The canola oil was employed because it has low viscosity and it is easily available, moreover its kinematic characteristics are well documented in the literature. Two cameras recorded the dynamics, one was placed above the channel and the other sideways. The experimental data were collected thanks to the use of a software developed by Birnbaum that extrapolates the information from the camera frames. Figure 3.14 shows one frame of a video recorded by the lateral camera: the yellow line on the left was set manually to represent the position of the dam, instead the blue line on the right is produced automatically by the software and refers to the advancing of the front.

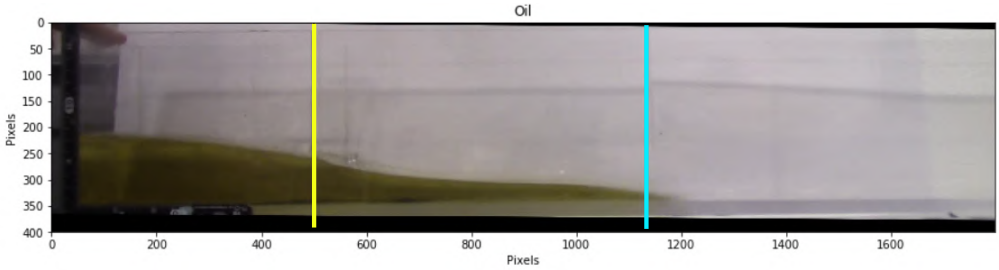


Figure 3.14: *Dam-break of a viscous fluid over a flat bottom.* Side view of the laboratory experiment with canola oil.

Our simulations tried to reproduce the laboratory experiment. The dam is $L = 0.2\,\mathrm{m}$ long and $h = 0.1\,\mathrm{m}$ high, the entire horizontal channel has length $L_{TOT} = 1.6\,\mathrm{m}$, while the kinematic viscosity of canola oil is $\nu = 5 \cdot 10^{-5}$. The domain is discretized with a cell-size

$\Delta x = 5 \cdot 10^{-3}$, and we used the PVR with the generalized minmod limiter ($\theta = 2$), the R-K scheme with 3 stages and the usual condition $k = 0.45$ for the CFL condition (3.28). Figure 3.15 shows the front advancement and compares the results of our simulations with those obtained with two laboratory experiments.

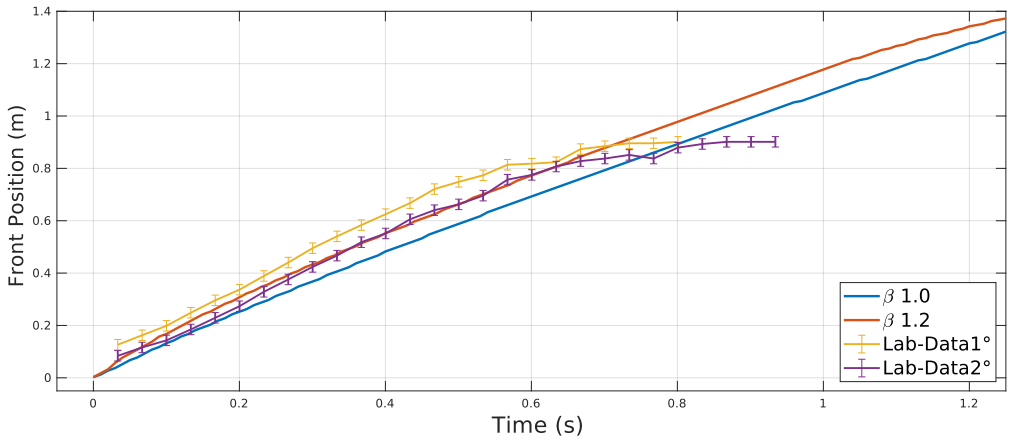


Figure 3.15: *Dam-break of a viscous fluid over a flat bottom.* Front advancing of the two laboratory experiments and of two different simulations, one obtained with the constant velocity profile ($\beta_u = 1.0$) and the other with the parabolic velocity profile ($\beta_u = 1.2$).

The results of the laboratory tests need some additional comments. After 0.6 s they present a slowdown, but such data are not reliable because they are related to the extreme of the visual field of the camera and hence are affected by several errors. Similarly, errors affect also the evolution at the initial time so that the data are not reported. Therefore we must limit ourselves to observe and compare results till 0.6 s and, from the plot of Figure 3.15, one can conclude that the simulation with the parabolic profile ($\beta_u = 1.2$) fits better the laboratory data. In summary, this comparison is incomplete: more tests had to be done and, if we could have continued to work in the laboratory, we would have had to make changes and use precautions to improve data collection: coloring the oil with a dark color to facilitate the software in identifying the front of the fluid, use a camera with a higher frame rate per second, and finally use two lateral cameras in order to have a correct identification of the front along the entire length of the channel. Since now the laboratory is open and functional, we hope to have the opportunity to conduct new experiments in the near future.

3.4.3 Dam-break of viscous fluids over an inclined bottom

The purposes of this test are analogous to those of the previous one: (*i*) check the correct treatment of wet/dry states and low/high viscosity; (*ii*) compare the solutions (by computing the relative difference with Eq.(3.27)) obtained with constant and parabolic velocity profile, PVR and CVR and 2 and 3 stages Runge-Kutta schemes, as reported in Tables 3.1–3.2; (*iii*) study the limiters effects.

The dam-break setting is like that of the former case §3.4.2, but the plane is inclined at a slope of 2.5° from horizontal. Unfortunately, to our knowledge, there is no reference solution with whom to compare our results.

High viscosity For the high-viscosity test we have considered again a silicone gum in isothermal condition, with a kinematic viscosity $\nu = \mu/\rho = 3.7\,\mathrm{m}^2/\mathrm{s}$ (as suggested in [52]).

After $t = 100\,\mathrm{s}$, for both the constant and parabolic velocity profiles, the relative difference between results computed with PVR and CVR is small ($\sim 10^{-4}$) and that calculated between solutions obtained with 2 and 3 R-K stages is even smaller ($\sim 10^{-7}$). In Figure 3.16 the high sensitivity to different limiters is very clear, and once more the generalized minmod limiter with $\theta = 2$ is confirmed to produce better approximations even with a less refined grid.

As in the previous test with high-viscosity conditions, the Froude number is very small $Fr \approx 0.023$. In such subcritical regime situation, the inertial forces are minor compared to gravitational and viscous forces and the effect of the vertical velocity profiles on flow dynamics is negligible. Indeed, results obtained with $\beta_u = 1$ or $\beta_u = 1.2$ are similar, with a relative difference of the order of 10^{-6}. In this test, the value of Fr is obtained again by using the front velocity propagation and the mean thickness over the flow extent, since the front thickness is comparable with the mean value.

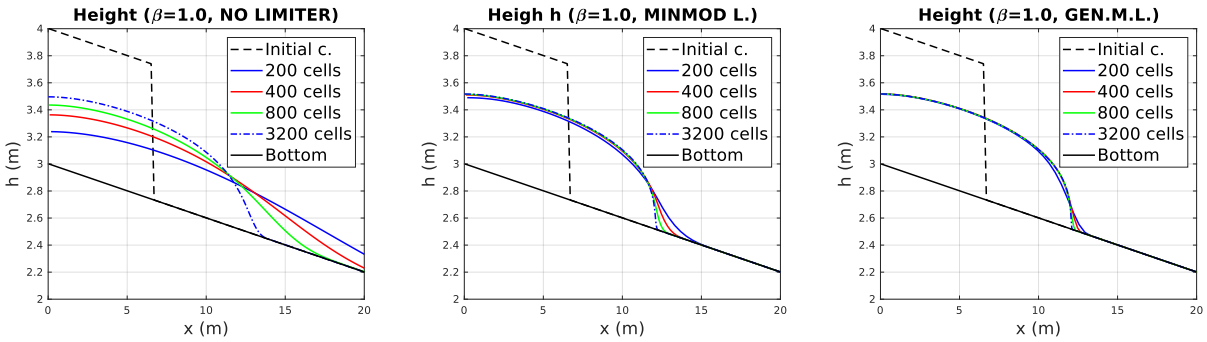


Figure 3.16: *Dam-break of a viscous fluid over an inclined bottom.* Solution at $t = 100$s for a high-viscosity fluid. Comparison between solutions computed with different grid size over a domain 75m long. *Left*: no limiter. *Center*: minmod limiter. *Right*: generalized minmod limiter.

Using a 400 cells grid, for the simulation with PVR, the generalized minmod limiter and $k = 0.45$ for the time step condition (3.28), we needed 3265 time steps with a total execution time of 7.95 s, whereas the time-step size grew up slowly and reached the maximum value of $3.4 \cdot 10^{-2}\,\mathrm{s}$.

Low viscosity For the low-viscosity test, we model again a silicone oil with the kinematic viscosity $\nu = \mu/\rho = 1.16 \cdot 10^{-3}\,\mathrm{m}^2/\mathrm{s}$ (taken from Cordonnier et al. [52]). Also for this test, temperature is not considered.

At $t = 10\,\mathrm{s}$, regardless of the assumption about the velocity profile, the relative distance between simulations computed with R-K scheme with 2 and 3 stages is about 10^{-3}; whereas, comparing CVR and PVR, the relative distance is one order of magnitude larger; moreover, there is no sensitivity to the limiter adopted.

For a fixed limiter and a fixed number of Runge-Kutta stages, there is a great difference in solutions with constant and parabolic velocity profile, $\beta_u = 1$ and $\beta_u = 1.2$ respectively, as shown in Figure 3.17. In particular, at $t = 5\,\mathrm{s}$ the runout of the simulation with $\beta_u = 1.2$ is larger than that obtained with $\beta_u = 1$, whereas at $t = 10\,\mathrm{s}$ the opposite occurs. The sensitivity to the velocity profile assumption finds correspondence in the supercritical regime situation, shown by the Froude number reported in Figure 3.17. Indeed, in such cases, after the initial collapse, the inertial force dominates the dynamics and then the effects of different values of β_u are considerable, as already observed in the former low-viscosity case in §3.4.2. In this case, the value of Fr is computed using the front velocity

of propagation and the flow thickness averaged over the 10 m close to the front, avoiding the thin tail.

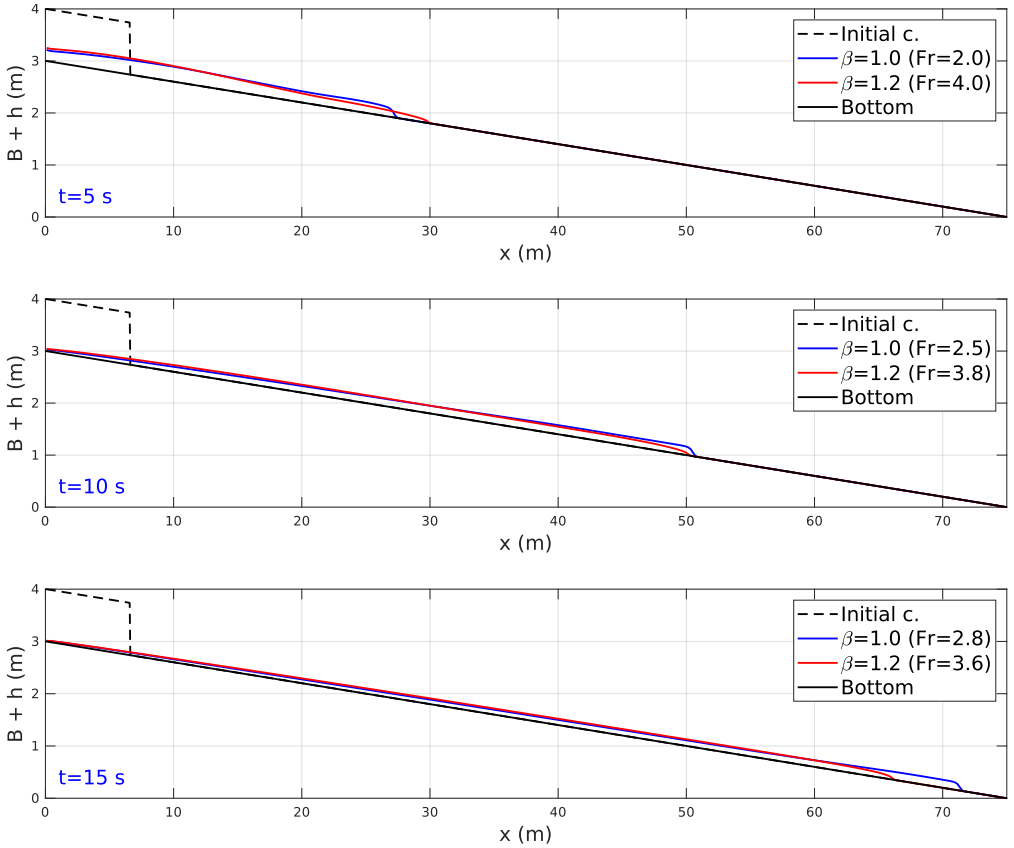


Figure 3.17: *Dam-break of a viscous fluid over an inclined bottom.* Evolution: *top* $t = 5\,\mathrm{s}$, *center* $t = 10\,\mathrm{s}$, *bottom* $t = 15\,\mathrm{s}$ of a low-viscosity fluid, computed by PVR at interfaces with the minmod limiter, 3 stages IMEX R-K scheme. Comparison between $\beta_u = 1.0$ and $\beta_u = 1.2$, i.e. constant and parabolic velocity profile. The Froude number Fr points out a supercritical regime in both conditions; its trend increases for $\beta_u = 1.0$, while it is decreasing for $\beta_u = 1.2$.

Furthermore, one observes that the tail does not move down the slope with time. This happens because of the rheological model adopted, with the friction force in Eq. (2.11) inversely proportional to h; for this reason, when the tail thickness decreases, the viscosity increases and hence the velocity tends to zero. Looking at the momentum equation in Eq. (3.1), it is also evident that on an inclined plane ($\nabla B \neq 0$), when a stationary condition with constant flow thickness is approached (i.e. when the terms on the right hand side of the momentum equation tend to zero), also flow thickness has to go to zero. This fact depends on the choice of a Newtonian rheology (see §1.1.5.7, where the friction force depends linearly on flow velocity. In a Binghamian rheology or a Voellmy-Salm rheology model (see de' Michieli Vitturi et al. [67] for details), where a yield slope term (velocity independent) is considered in the friction forces, a critical thickness for which flows do not move on an inclined plane may be found.

Using a 400 cells grid, the simulation with PVR, generalized minmod limiter and with $k = 0.45$ at the time step condition (3.28), in the case of $\beta_u = 1$ (constant velocity profile) took 751 time steps with a total execution time of 2.65 s whereas the time-step size has grown up to $1.4 \cdot 10^{-2}$s; in the case of $\beta_u = 1.2$ (parabolic velocity profile) the simulation took 972 time steps with a total execution time of 2.65 s whereas the time-step size grew

increased till $1.2 \cdot 10^{-2}$s. See Table 3.6 on page 158 that reports this information and that of the other tests.

3.4.4 BM2: Inclined viscous isothermal spreading

The origin of this 2D benchmark test comes from a laboratory experiment seeing silicon oil spreading on an inclined plane of slope α, injected through a point hole at a constant flow rate Q as represented in Figure 3.18. Such experiment was used by Lister [172] to test numerical results and furthermore he derived analytical solution for this, and nowadays it is considered an important benchmark to test numerical models used on a simple geometry. The set-up parameters we have used follow those used by Lister [172]: the source point has circular area with radius $r = 10^{-3}$ m, the plane is inclined at a slope of $\alpha = 2.5°$ from horizontal, the fluid supply rate is $R = 1.48 \cdot 10^{-6}\,\mathrm{m^3\,s^{-1}}$ and the kinematic viscosity is $\nu = 11.3 \cdot 10^{-4}\,\mathrm{m^2\,s^{-1}}$; a Newtonian viscosity is assumed and none thermal phenomena accounted.

For our simulation, we considered a rectangular domain $[-35, 115] \times [-45, 45]$, where all the lengths are expressed in cm, and we discretized it with two different grid resolutions: the rough discretization has 150×90 cells, the finer one has 300×180 cells. We used the PVR with the generalized minmod limiter ($\theta = 1.3$) and a 3 stages IMEX R-K scheme for the time discretization (see Table 3.2), with $k = 0.24$ for the time step condition (3.28). In every plot reported, we consider only values of h greater than 10^{-3} which is rather reasonable since they correspond to the 95% of the total values computed (in particular, the averaged value of h is $1.44 \cdot 10^{-3}$).

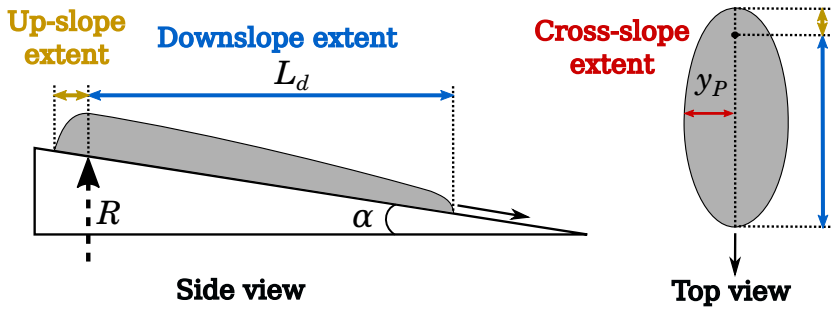


Figure 3.18: *BM2: Inclined viscous isothermal spreading.* Set up of the benchmark.

Figure 3.19 reports the flow contours as a function of time. On the left there are the results presented in Lister [172] that come from laboratory experiment and numerical simulation, while on the right there are the results obtained with our numerical model with the fine spatial discretization. It is immediate to note that, for all results represented, the up-slope extent rapidly reaches a steady state, whereas the flow continues to advance down-slope with time.

We extrapolate from the laboratory results (namely from the original graph of Lister [172] that is pictured in Figure 3.19) the data of the down-slope and cross-slope extents because they are useful for the arguments that follow; we report them in Table 3.3 and denote them as L_d and y_P respectively.

Lister determined a characteristic time t^* that separates two different behaviours of the spreading. Initially, for the so said “short time", fluid spreads radially from the source, as it would be on a horizontal plane, because the thickness h at the interfaces is much greater than that of the inclined plane, after that the opposite occurs for the so said “long

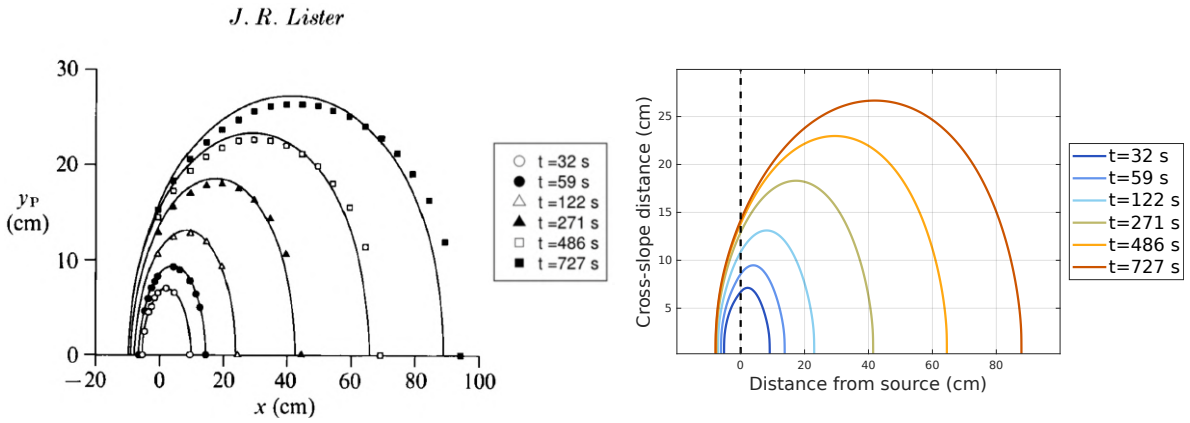


Figure 3.19: *BM2: Inclined viscous isothermal spreading.* Representation at various times of the flow front. *Left:* results from Lister [172], where lines describe the evolution of laboratory experiment while symbols the predicted results obtained from numerical simulations. *Right:* results obtained with our code with the high-resolution discretization; by "front" we mean the position where h becomes less than 10^{-3}.

Table 3.3: *BM2: Inclined viscous isothermal spreading.* Down-slope L_d and cross-slope y_P extents extrapolated from the laboratory experiment results (Figure 3.19 on the left) that come from [172].

time (s)	32	59	122	271	486	727
L_d (cm)	9.5	14.5	23.5	42.5	65.5	88.5
y_P (cm)	7.0	9.4	13.0	18.4	23.3	27.2

time", and the flow is predominantly down-slope, with some cross-slope spreading. This behaviour is due to the pressure terms, see Eq. (2.10), that can be rearranged as

$$\int_B^{B+h} \nabla p(z) dz = \nabla \left(\frac{1}{2} \rho g h^2 \right) + \rho g h \nabla B = \rho g h \nabla h + \rho g h \nabla B,$$

for the short time, the term with ∇h is bigger than the other, whereas, for the long time, the term with ∇B becomes dominant and then the flow follows mostly the topography. The first situation is referred to as "density current", the second is named "avalanche" [67].

The characteristic time depends on the physical parameters of the test:

$$t^* = \left[\frac{(\cot \alpha)^5}{R} \left(\frac{3\nu}{g \sin \alpha} \right)^3 \right]^{1/4}, \tag{3.31}$$

and in our case its value is ≈ 38 s. Figure 3.20 presents the evolution of the fluid extent in the three directions, namely down-slope, cross-slope and up-slope, to confirm that also the dynamic of our simulation is almost symmetric before the characteristic time.

The convergence study for the down-slope and cross-slope extents is presented in Figures 3.21 and 3.22: the results of our simulations obtained with the high and low-resolutions are compared with the data collected from the laboratory experiments (see Table 3.3). One observes that our simulations have a good agreement with the experimental results.

The simulations presented for this benchmark were obtained with a parabolic velocity profile adoption ($\beta_u = 1.2$), but the assumption of constant velocity profile ($\beta_u = 1.0$)

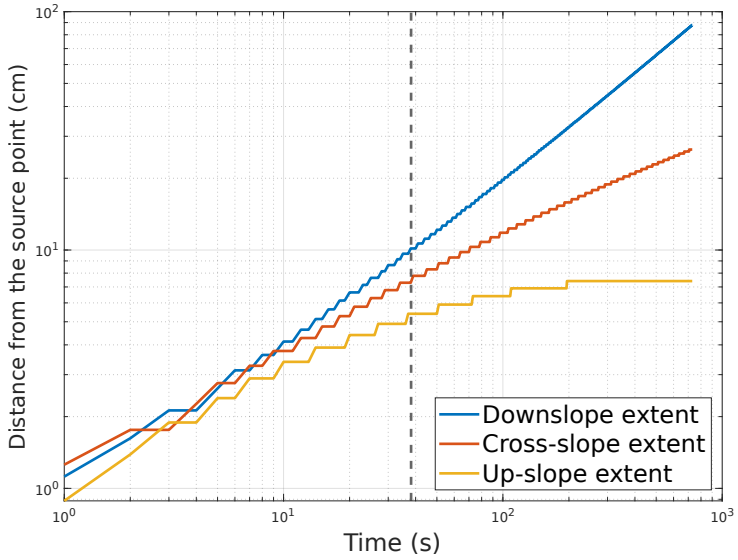


Figure 3.20: *BM2: Inclined viscous isothermal spreading.* Time evolution of the down-slope extent, the cross-slope propagation and the up-slope extent. The dashed line refers to the characteristic time t^*, defined in Eq. (3.31). The results illustrated were computed with the high-resolution spatial discretization.

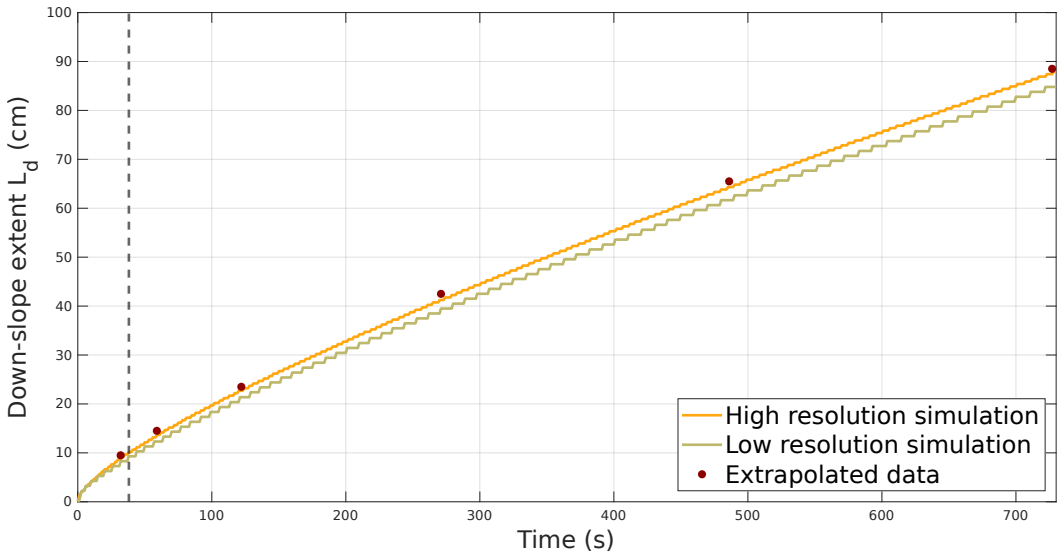


Figure 3.21: *BM2: Inclined viscous isothermal spreading.* Time evolution of the down-slope extent $L_d(t)$. The extrapolated data refer to Table 3.3. The dashed line refers to the characteristic time t^*, defined in Eq. (3.31).

produces similar results. The maximum value reached by h is $5 \cdot 10^{-3}$ m, while the relative distance between solutions obtained with different velocity profiles is about 10^{-6} m, therefore the relative distance between the two solutions is of 1%, which is quite negligible. This behaviour is not surprising, in fact, even though the fluid has a low viscosity, it moves very slowly and the dynamics is driven by the gravity force and not by inertia.

Lister derived the asymptotic scaling behaviors of the flow front advance $L_d(t)$ and of

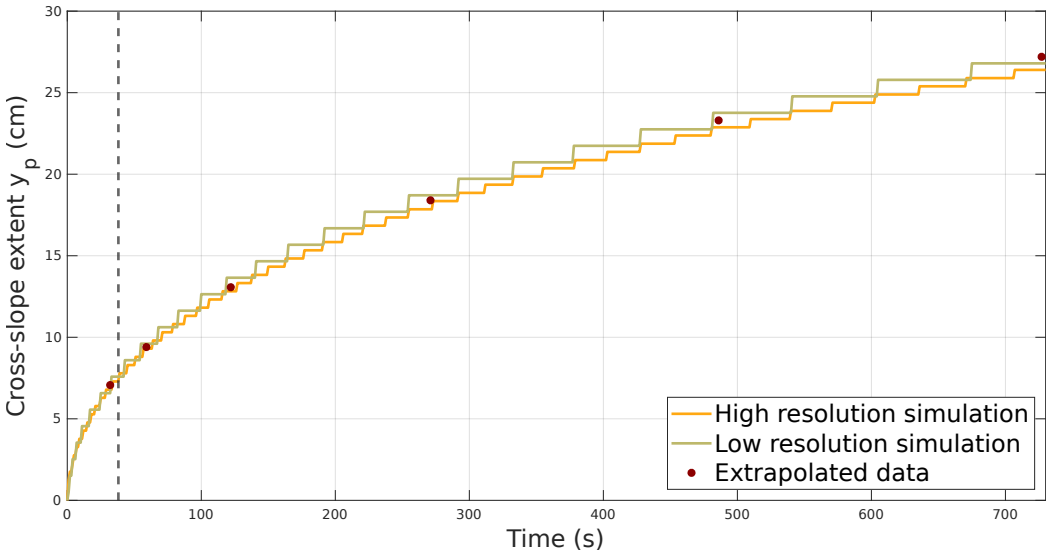


Figure 3.22: *BM2: Inclined viscous isothermal spreading.* Time evolution of the cross-slope extent $y_P(t)$. The extrapolated data refer to Table 3.3. The dashed line refers to the characteristic time t^*, defined in Eq. (3.31).

the cross-slope extent $y_P(t)$ for the "long time" dynamics:

$$L_d(t) \sim \left[\left(\frac{g}{3\nu}\right)^3 \frac{R^4 \sin^5 \alpha}{\cos^2 \alpha} \right]^{1/9} t^{7/9}, \quad t \gg t^* \tag{3.32a}$$

$$y_P(t) \sim \left(\frac{R \cos \alpha}{\sin \alpha} \right)^{1/3} t^{1/3}, \quad t \gg t^*. \tag{3.32b}$$

In an effort to compare our results with the theoretical functions derived by Lister [172] Eqs. (3.32), we search for a qualitative agreement with the dimensionless extents. We use the characteristic time t^* defined in Eq. (3.31) and the characteristic lengths defined as

$$x^* = y^* = \left(\frac{R \cos^3(\alpha) 3\nu}{g \sin^4(\alpha)} \right)^{\frac{1}{4}} \approx 10.9 \text{ cm}. \tag{3.33}$$

In Figure 3.23 both the theoretical solutions and our simulation results are represented, but they compare as dimensionless variables computed as:

$$T := \frac{t}{t^*}, \quad X := \frac{L_d}{x^*}, \quad Y := \frac{y_P}{y^*}. \tag{3.34}$$

The theoretical and the experimental results show a qualitative agreement for the last times, confirming a decent asymptotic agreement. However, not negligible quantitative differences rise and there are not enough elements in the work of Lister that help us to understand the reasons behind them.

For the high-resolution simulation, where we used 300×180 cells, we needed 144165 time steps with a total execution time of 60170 s, whereas the time-step grew up quite quickly and reached the maximum value of 0.01 s and then decreased slowly settling at a constant value 0.005 s (see Table 3.6 on page 158 that reports this information and that of the other tests).

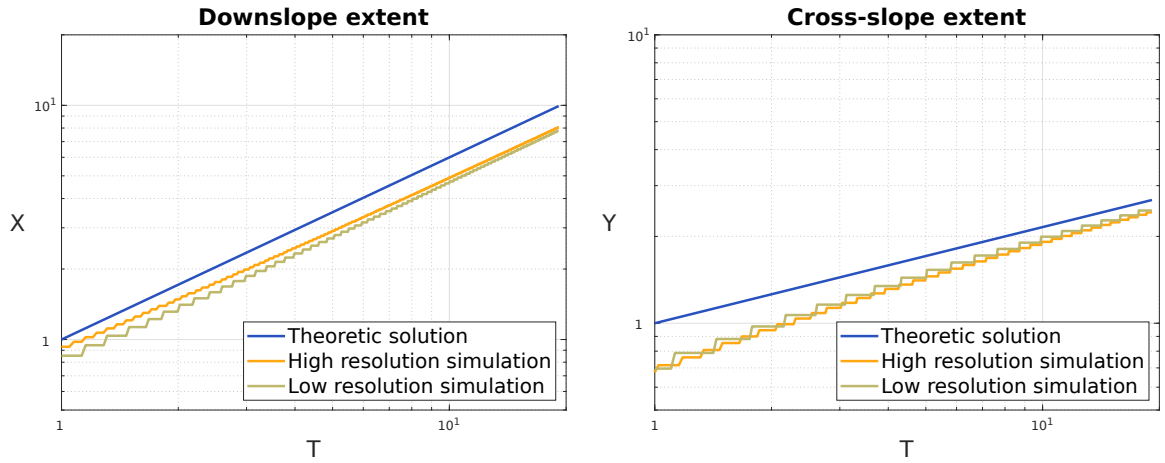


Figure 3.23: *BM2: Inclined viscous isothermal spreading.* Time evolution of the dimensionless downslope X and cross-slope extent Y, as defined in Eq. (3.34), against the dimensionless time T. Only the "long time" results are represented.

3.4.5 Temperature-dependent viscosity

This test is a 2D simulation of a temperature-dependent viscous fluid. Our aim is to check the coupling between momentum and temperature equations of the system (2.40), and the effects of a Newtonian rheology.

The fluid is initially concentrated in a hemispherical shape over a 35° inclined flat plane, then it slides down over such plane which merges continuously into a horizontal plane, passing through a smooth transition zone. The initial conditions and the topography of this test are similar to that used in Example 4.1 from Wang et al. [269]. The computational domain is a rectangle 30 m long and 20 m wide and the inclination angle is defined as

$$\alpha(x) = \begin{cases} 35^\circ, & 0 \leqslant x < 17.5, \\ 35^\circ(1 - (x - 17.5)/4), & 17.5 \leqslant x < 21.5, \\ 0^\circ, & 21.5 \leqslant x \leqslant 30. \end{cases}$$

In this test we use viscosity and temperature values representative of lava flows. We also consider a temperature-dependent viscosity in the last term of the momentum equation (2.40b), adopting an exponential relationship as the one suggested by Costa and Macedonio [55]

$$\gamma = \frac{3\,\nu(T)}{h}, \quad \nu(T) = \nu_r \exp[-b(T - T_{ref})],$$

where b is an appropriate rheological parameter and ν_r is the reference kinematic viscosity that the fluid has at the reference temperature T_{ref}. Initially, the depth-averaged temperature and the reference temperature are set equal ($T = T_{ref} = 1353$ K), hence viscosity coincides with the reference viscosity.

The domain is discretized by 80×54 cells. The PVR is used at the interfaces with the generalized minmod limiter ($\theta = 1.3$) and with a 3 stages IMEX R-K scheme for time marching, see Table 3.2. For the characteristic thermal boundary-layer (see §2.1.3), we adopt two thickness values $\delta_T = h/n$ with $n = 4$ and $n = 10$, which lead to different values for β_T:

$$\beta_T = \begin{cases} 0.234375, & n = 4, \\ 0.0975, & n = 10. \end{cases}$$

One reminds that the presence of non trivial values for β_T and β_{T_1} depends on the fact that we have considered a parabolic velocity profile, therefore in these cases $\beta_u = 1.2$ must be set, otherwise the model would lose its consistency.

High viscosity We have adopted again the value $\nu_r = 3.7\,\mathrm{m}^2/\mathrm{s}$ (taken from Example 1, Cordonnier et al. [52]) for the kinematic viscosity. In Figure 3.24 we compare, for $n = 4$, results obtained at time $t = 600\,\mathrm{s}$ with constant and variable temperature profiles, respectively, while in Figure 3.25 we present the time evolution of their contour. For this viscosity value, the differences between solutions obtained with $n = 4$ or 10 are negligible, thus the results of the latter case are not shown.

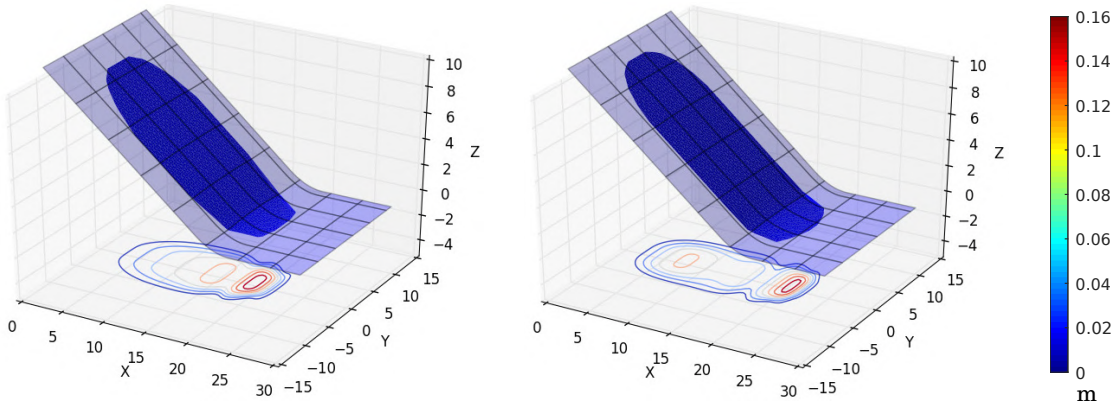


Figure 3.24: *2D Simulation of a temperature-dependent viscous fluid.* Comparison of high-viscosity fluid simulations computed with different thermal profiles, at $t = 600\,\mathrm{s}$: *left* constant with $T_1 = 1353\,\mathrm{K}$, *right* piecewise linear with $T_1 = 1400\,\mathrm{K}$.

When a constant temperature profile is assumed ($T_1 = T = T_{ref}$), T is simply advected with the flow and the kinematic viscosity equals the reference value ν_r. If the piecewise linear temperature profile is considered, the maximum temperature is set higher than the averaged temperature ($T_1 = 1400\,\mathrm{K}$) and then T changes (both in space and in time) during the simulation. In the latter case, due to velocity and thermal profiles, the top layer has the highest temperature and moves with a speed larger than the lower part, hence it propagates faster arriving soon to constitute the front; on the contrary, the fluid at the upper part of the plane becomes, on average, colder. In this situation, at time $t = 600\,\mathrm{s}$, the depth-averaged temperature reaches the minimum $T = 1247\,\mathrm{K}$ at the upper part of the plane, approximately at $x \approx 4.5\,\mathrm{m}$; such minimum leads to a kinematic viscosity value higher than the reference one, and in this situation an amount of fluid is accumulated at the upper part of the plane, whereas in the case of constant thermal profile this little pile does not remain.

For 600 s of simulation, with $k = 0.24$ for the time step condition (3.28), in the constant temperature profile case the computation needed 9583 time steps, a total execution time of 547.06 s, while the time-step size reached the maximum value of $7.3 \cdot 10^{-2}\,\mathrm{s}$. Whereas, the case of piecewise linear temperature profile needed 8804 time steps, a total execution time of 522.83 s, and the time-step size reaches the maximum value of $7.9 \cdot 10^{-2}\,\mathrm{s}$.

Low viscosity For the low-viscosity test, we used again the value $\nu_r = 1.16 \cdot 10^{-3}\,\mathrm{m}^2/\mathrm{s}$ for the kinematic viscosity (taken from Cordonnier et al. [52]).

We start comparing solutions obtained by different characteristic thermal boundary layers δ_T/n, with $n = 10$ and $n = 4$, and with the maximum temperature on the surface

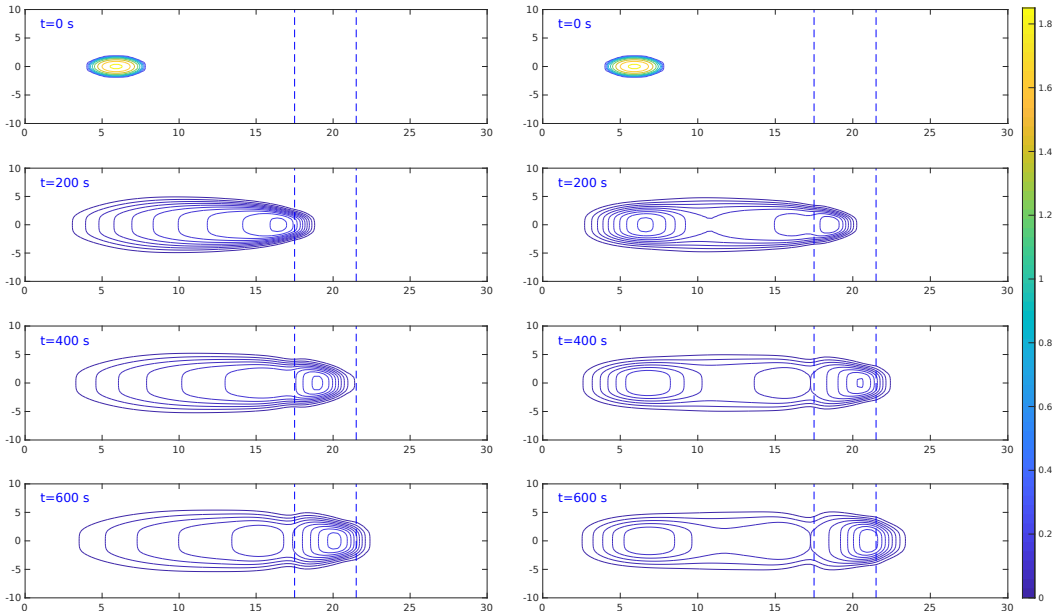


Figure 3.25: *2D Simulation of a temperature-dependent viscous fluid.* Evolution of high-viscosity fluid simulations computed with different thermal profiles: *left* constant profile with $T_1 = 1353\,\mathrm{K}$, *right* piecewise linear profile and $T_1 = 1400\,\mathrm{K}$. The section between the two dotted lines is the transition zone from the inclined plane to the horizontal plane.

Table 3.4: *2D Simulation of a temperature-dependent viscous fluid.* Performance of the solver for 600 s of simulation of the high viscosity case.

Test			Performance			
δx (m)	cells	ν (m^2/s)	execution time (s)	time step	velocity profile	temperature profile
0.375	80×54	3.7	522.83	8804	parabolic	piecewise ($n = 4, 10$)
			547.06	9583	parabolic	constant

as $T_1 = 1400\,\mathrm{K}$, see Figure 3.26. When $n = 10$ the fluid propagates faster because the top layer with the maximum temperature is thicker, therefore the averaged temperature is greater and the kinematic viscosity lower. However, as in the former high-viscosity case, there is a little pile of matter in the upper part of the plane. In addition, in both cases, at the last time steps there is a loss of mass from the computational domain since the fluid reaches the boundary and goes on propagating outside. The computation of 3 s of simulations in both cases of $n = 4$ and $n = 10$, with $k = 0.24$ for the time step condition (3.28), required 552 time steps and 28.89 s, while the time step size reached the maximum value $1.1 \cdot 10^{-2}$ s.

In order to underline how big may be the difference in considering or not a temperature dependent viscosity, we conclude with simulations with a temperature independent viscosity and a simple transport of temperature. We compare again solutions with different velocity profiles (as in the previous 1D tests §3.4.2, §3.4.3). A constant temperature profile and a constant kinematic viscosity are assumed by setting $T_1 = T = T_{ref} = 1353\,\mathrm{K}$. We find, once more, that with a constant profile the fluid propagates faster and the front is thicker, see Figure 3.27. For 3 s of simulation, with $k = 0.24$ for the time step condition (3.28), the case of a parabolic velocity profile needed 502 time steps with a total execu-

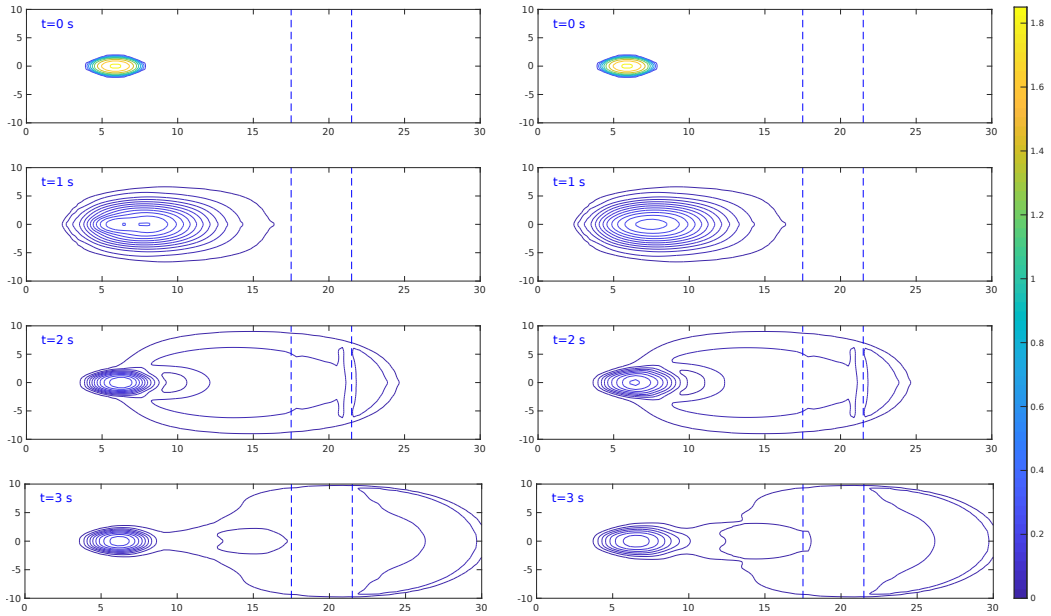


Figure 3.26: *2D Simulation of a temperature-dependent viscous fluid.* Evolution of a low-viscosity fluid with temperature-dependent viscosity computed with different characteristic thermal boundary layer $\delta_T = h/n$: *left* $n = 10$, *right* $n = 4$. The section between the two dotted lines is the transition zone from the inclined plane to the horizontal plane.

tion time of 24.18 s, whereas the time-step size has reached the maximum value of 10^{-2} s; instead, the case of a constant velocity profile took 353 time steps with a total execution time of 16.69 s, whereas the time-step size has reached the maximum value of $1.2 \cdot 10^{-2}$ s. See Table 3.6 on page 158 that reports this information and that of the other tests.

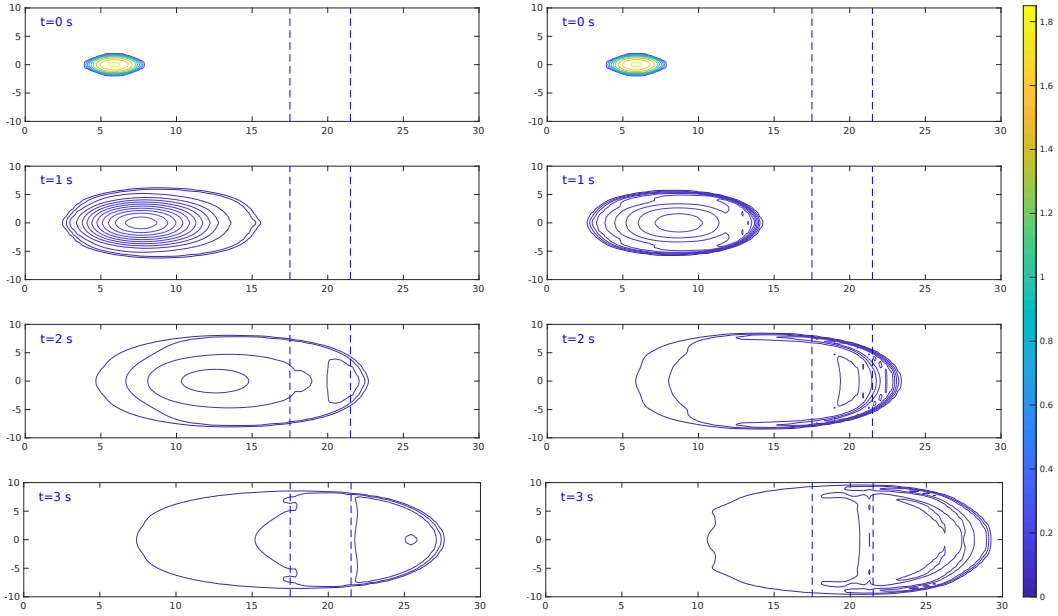


Figure 3.27: *2D Simulation of a temperature-dependent viscous fluid.* Evolution of a low-viscosity fluid with viscosity not temperature dependent with different vertical velocity profile: *left* parabolic profile, hence $\beta_u = 1.2$, *right* constant profile, hence $\beta_u = 1.0$. The section between the two dotted lines is the transition zone from the inclined plane to the horizontal plane.

3.4.6 BM3: Axisymmetric cooling and spreading

This benchmark concerns the spreading of a warm viscous fluid, onto a flat plane, which cools down during the dynamics due to heat exchanges with the environment. This is the intermediate test proposed by Cordonnier et al. [52] between the spreading-only benchmarks, such as BM1 and BM2, and the most complex lava flow case, that follows. The viscosity is still assumed Newtonian and temperature-independent, therefore there is no relation between rheology and thermal structure.

This test refers to an analog experiment reported in Garel et al. [99], denoted as C14, where a hot silicone oil (Rhodorsil® 47V 5000 or 47V 12500, dyed red) is injected, at a constant supply rate $R = 2.2 \cdot 10^{-8}[\mathrm{m^3 s^{-1}}]$, onto a horizontal plane of polystyrene from a point source of 2–4 mm of radius. Table 3.5 shows the values of the physical parameters involved into the experiment, most of them are reported in Cordonnier et al. [52], the others are available only on the original paper Garel et al. [99]. There is a difference in the density value indicated in the two papers, so we adopted the value reported in the original work Garel et al. [99].

Table 3.5: *BM3: Axisymmetric cooling and spreading.* Physical parameters of the hot silicon oil simulation. The symbol * refers to the parameters available in Garel et al. [99] and not provided in Cordonnier et al. [52].

Symbol	Value	Definition	Unit
ρ	954	fluid density	$\mathrm{kg\,m^{-3}}$
μ	3.4	dynamic viscosity	$\mathrm{Pa\,s}$
c_p	1500	specific heat of fluid	$\mathrm{m^2\,s^{-2}\,K^{-1}}$
k	0.15	thermal conductivity of fluid	$\mathrm{W\,m^{-1}\,K^{-1}}$
κ	10^{-7}	thermal diffusivity of fluid *	$\mathrm{m^2\,s^{-1}}$
λ	2	convective heat transfer coeff.	$\mathrm{W\,m^{-2}\,K^{-1}}$
ϵ	0.96	emissivity	-
T_{env}	293.15	temp. of environment	K
T_{vent}	315.15	temp. of fluid at the vent	K
T_{soil}	293.15	temp. of soil	K
k_{soil}	0.03	thermal conductivity of soil *	$\mathrm{W\,m^{-1}\,K^{-1}}$
κ_{soil}	$6 \cdot 10^{-7}$	thermal diffusivity of soil *	$\mathrm{W\,m^{-2}\,J^{-1}}$
R	$2.2 \cdot 10^{-8}$	effusion rate	$\mathrm{m^3 s^{-1}}$

In this simulation the system of equations (2.42) is solved, coupled with the condition of constant density. The square $[-12, 12] \times [-12, 12]$, where all the lengths are expressed in cm, determines the domain, discretized with two different grid resolutions, a coarse one with $\Delta x = 2 \cdot 10^{-3}$ and a fine one with $\Delta x = 10^{-3}$. The circular vent of the experiment is located in the center of the domain and it is approximated by the squares of the discretization grid. From the modeling point of view, we adopted the parabolic velocity profile, with $\beta_u = 1.2$, and the piecewise linear temperature profile, with β_T defined in Eq. (2.30). The parameters introduced in §2.1.5 related with the temperature are: $n = 4$ and $M = 12$ for the thermal boundary layers of the oil and of the polystyrene surface respectively, the fraction of the exposed inner core $f = 1$ because the fluid is completely melt (see §2.1.5.3), the rheology parameter of Eq. (2.13) is $b = 0$. For the numerical schemes, the PVR is used for the interfaces with the generalized minmod limiter ($\theta = 1.3$) and the 3 stages IMEX R-K scheme (reported in Table 3.2) for the time marching.

The dynamics of the experiment is completely symmetric with a "radial" flow advance described by the analytic expression

$$x(t) \approx 0.715 \left[\frac{gR^3}{3\nu} \right]^{1/8} t^{1/2}, \tag{3.35}$$

determined in the theory of Huppert [131]. We use the analytic expression of $x(t)$ for a convergence study with the high and low resolution simulations, see Figure 3.28. The threshold used to define the fluid front is the thickness $h = 10^{-4}$ m, differently from the other tests where it was $h = 10^{-3}$ m. Such different choice is related to the particularly tiny thickness of this test: as shown in Figure 3.29, neglecting thicknesses less than a millimeter means to completely neglect what happens in the first 20 seconds and to neglige an important part of the fluid for the rest of time.

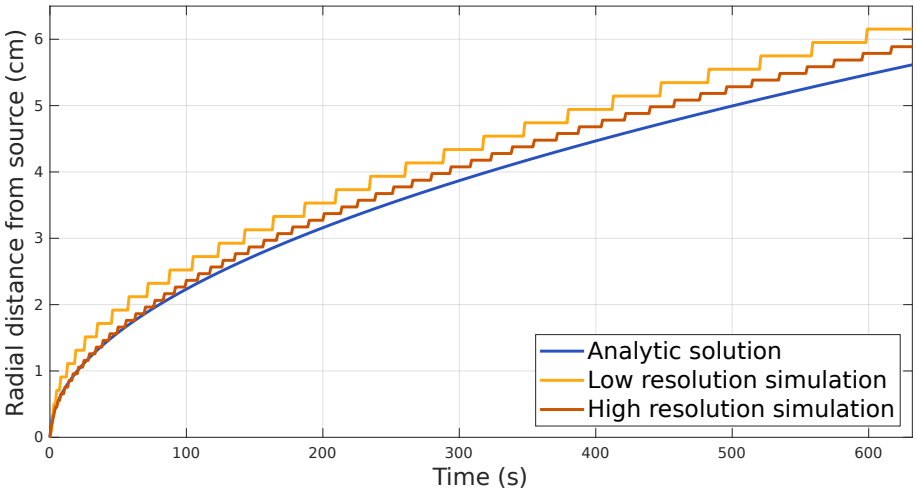


Figure 3.28: *BM3: Axisymmetric cooling and spreading.* Convergence study, comparing the theoretical front position, Eq. (3.35), with the results of simulations obtained with low and high resolution grids.

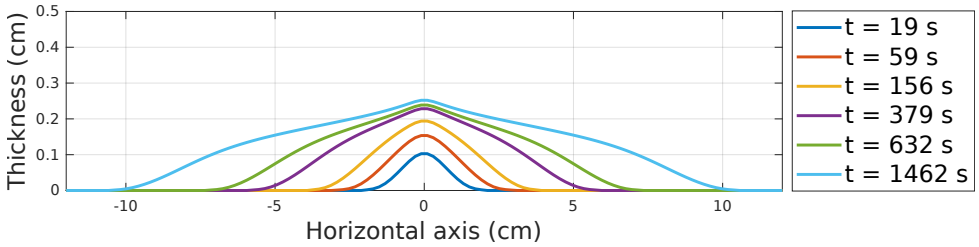


Figure 3.29: *BM3: Axisymmetric cooling and spreading.* Thickness of the simulated fluid at different time.

Figure 3.28 highlight the good convergence of the solutions computed with the simulations to the analytic solution. For a further investigation, we computed the simulations by setting the constant velocity profile (i.e. $\beta_u = 1.0$). The results of the front extent do not change, conversely from what happened in the BM1 test case, §3.4.2. The reason is that, despite the low viscosity of the oil (which is of the same order of magnitude as the low viscosity case of BM1 test, §3.4.2), the velocity is so slow that the dynamics are driven by gravity force, instead of inertial force, resulting in a subcritical regime.

Figure 3.30 exhibits the time evolution of the surface temperature, in particular the dimensionless surface temperature is accounted, namely $(T - T_{env})/(T_{vent} - T_{env})$, and the figure reports both the graph with the results of our simulations (right) and the original graph with the laboratory results of Cordonnier et al. [52] (left). Our graph represents the surface temperature of the oil obtained with our simulation, while the original graph in Cordonnier et al. [52] reports the temperatures detected by an infrared camera located over the plane that takes measurements of both oil and plate that is not covered yet of oil. The big difference between the two graphs is the "tail" that appears in the first three-measurements in the original graph and that in our graph is completely missing. Such "tail" is due to the natural phenomenon of the thermal diffusion for which the plate and the air close to the oil are warmed by it resulting in a temperature higher than the ambient.

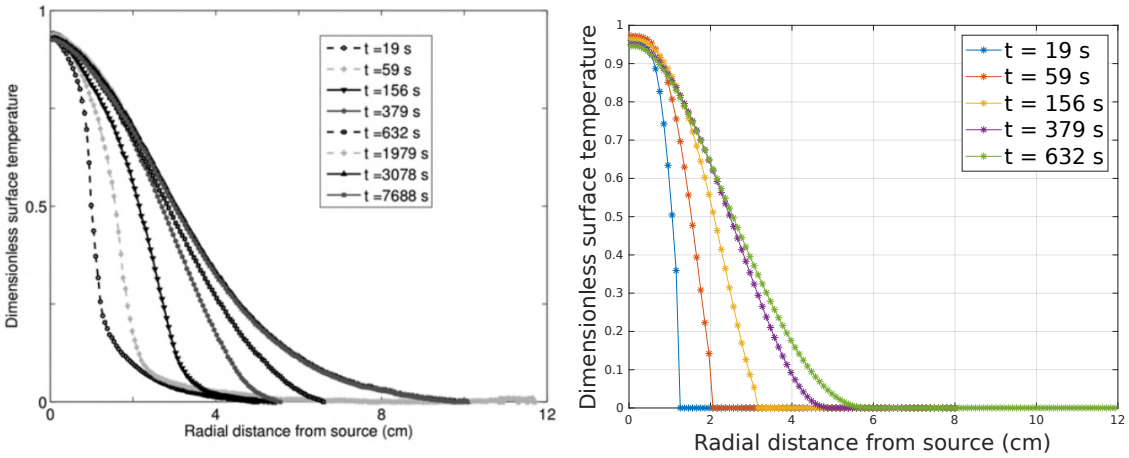


Figure 3.30: *BM3: Axisymmetric cooling and spreading.* Evolution in time of the dimensionless surface temperature profile, defined as $(T - T_{env})/(T_{vent} - T_{env})$. *Left:* original graph from the benchmarks article Cordonnier et al. [52]. *Right:* results of our model.

In Figure 3.31 is shown a further comparison with our results about the surface temperature with experimental and theoretical temperatures of the original paper Garel et al. [99]. The temperature is again represented normalized, and the same is for the radial distance from the source, therefore $x = 1$ corresponds to the fluid front. The agreement of our results is better with the experimental outcomes than with the theory.

The computation of 632 s of simulations with the high resolution grid, $\delta x = 10^{-3}$ m, took a total execution time of 96653.57 s, it required 409432 time steps and the time step size stabilized around 0.0015 (see Table 3.6 on page 158 that reports this information and that of the other tests).

3.4.7 Natural case: the Pico do Fogo 2014–2015 Eruption

In this test, we apply our model to the conditions of real effusive eruption. As a reference case, we consider the Fogo volcano at Cape Verde, West Africa, and its last eruption started on 23 November 2014 and ended on 8 February 2015. We use the actual topography of Fogo and the data characteristic of such event, as the vent location, the effusion rate, and the effusive temperature, to perform simulations of the first day of the eruption. The present test is not meant to try to reproduce as best as possible to real event but to analyze the rheological parameters' impact on the lava flow emplacement. Once

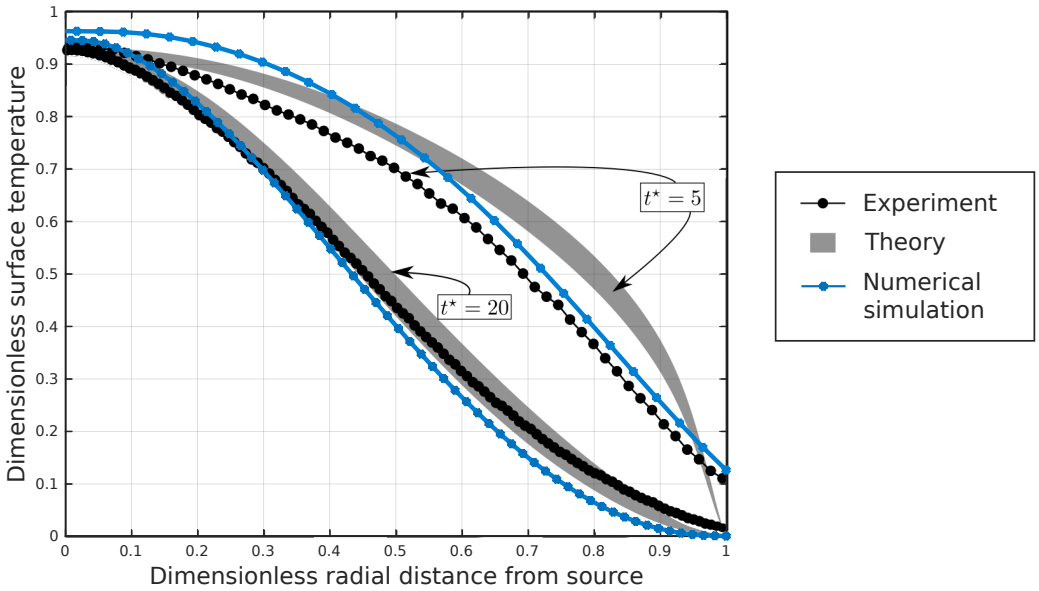


Figure 3.31: *BM3: Axisymmetric cooling and spreading.* Comparison of experimental, theoretical and simulated normalized surface temperatures, defined as $(T - T_{env})/(T_{vent} - T_{env})$, over the normalized radial distance. $t^* = 5$ corresponds to $t = 156\,\text{s}$ and $t^* = 20$ to $t = 620\,\text{s}$. In the original work Garel et al. [99], t^* is defined as $t^* = \dfrac{t}{\tau}$ where $\tau = \dfrac{0.715^{\frac{4}{3}}}{\kappa}\left(\dfrac{3\mu R}{\rho g}\right)^{\frac{1}{2}}$.

Table 3.6: The Table reports the performances of the solver for each previous test in a little selection of cases associated with the parameters written. Test 1: Riemann problem with discontinuous bottom, §3.4.1. Test 2: BM1, dam-break of viscous fluids over a flat bottom, §3.4.2. Test 3: Dam-break of viscous fluids over an inclined bottom, §3.4.3. Test 4: BM2, inclined viscous isothermal spreading, §3.4.4. Test 5: temperature-dependent viscosity, §3.4.5. Test 6: BM3, axisymmetric cooling and spreading, §3.4.6.

	Test parameters						Performance	
test	Δx (m)	cells	ν (m^2/s)	velocity profile	temp. profile	simulated time (s)	execution time (s)	time steps
1	0.02	600	0	constant	no temp.	0.5	2.26	473
2	0.1875	400	3.7	constant	no temp.	500	7.91	3432
2	0.1875	400	$1.16\cdot10^{-3}$	constant	no temp.	20	1.9	597
2	0.1875	400	$1.16\cdot10^{-3}$	parabolic	no temp.	20	2.44	757
3	0.1875	400	3.7	parabolic	no temp.	100	7.95	3265
3	0.1875	400	$1.16\cdot10^{-3}$	constant	no temp.	100	2.65	751
3	0.1875	400	$1.16\cdot10^{-3}$	parabolic	no temp.	100	2.65	972
4	0.5	300×180	$11.3\cdot10^{-4}$	parabolic	no temp.	727	60170.01	144165
5	0.375	80×54	3.7	parabolic	piecewise	600	522.83	8804
5	0.375	80×54	3.7	parabolic	constant	600	547.06	9583
5	0.375	80×54	$1.16\cdot10^{-3}$	parabolic	piecewise	3	28.89	552
5	0.375	80×54	$1.16\cdot10^{-3}$	parabolic	constant	3	24.18	502
5	0.375	80×54	$1.16\cdot10^{-3}$	constant	constant	3	16.69	353
6	0.001	160×160	$3.56\cdot10^{-3}$	parabolic	piecewise	632	96653.57	409432

we obtained and analyzed the results of this sensitivity analysis, we choose rheological parameters that produce a result with a good fit with the real event. By keeping fixed such parameters, we investigate the impact of other factors on the simulation, namely: (*i*) different vent positions; (*ii*) different grid resolutions; (*iii*) different viscosity models. Finally, by using the same choice of rheological parameters, we simulate two days of eruption and compared the result with the emplacement of the real event.

Cape Verde Archipelago is located west of the Western Atlantic coast of Africa and has volcanic origins [65], see Figure 3.32. Fogo Island rises between Brava and Santiago islands and is the fourth largest island of the archipelago and the highest with 2829 m above the sea level of Pico do Fogo. An active volcano stands in the island's center, presents a 9 km wide caldera, Chã das Caldeiras ("Plain of the Calderas"), and has a summit at Pico do Fogo. An enormous crater rim, called Bordeira and up to 1 km high, encircles the caldera on its western side. Fogo volcano is the youngest and most active volcano of the archipelago [59, 71]. A violent eruption occurred in 1680, which was remembered because it could be admired even from hundreds of kilometers. During this eruption, the island took the actual name Fogo (that means "Fire"). In the twentieth century, only two effusive eruptions occurred (1951 and 1995), and in both cases lava invaded Chã das Caldeiras threatening and damaging farmings and the villages present of Bangaeira, Portela, and Ilhéu de Losna. The penultimate last but one eruption in 1995 formed a new crater called Pico Pequeno and developed an eruptive fissure close to that. In the last eruption, which began in November 2014 and lasted until February 2015, the eruptive fissure where the eruption started was on the southwest flank of Pico do Fogo, and it was almost in the same position as the fissure of the previous eruption of 1995, which is a rare behavior [239].

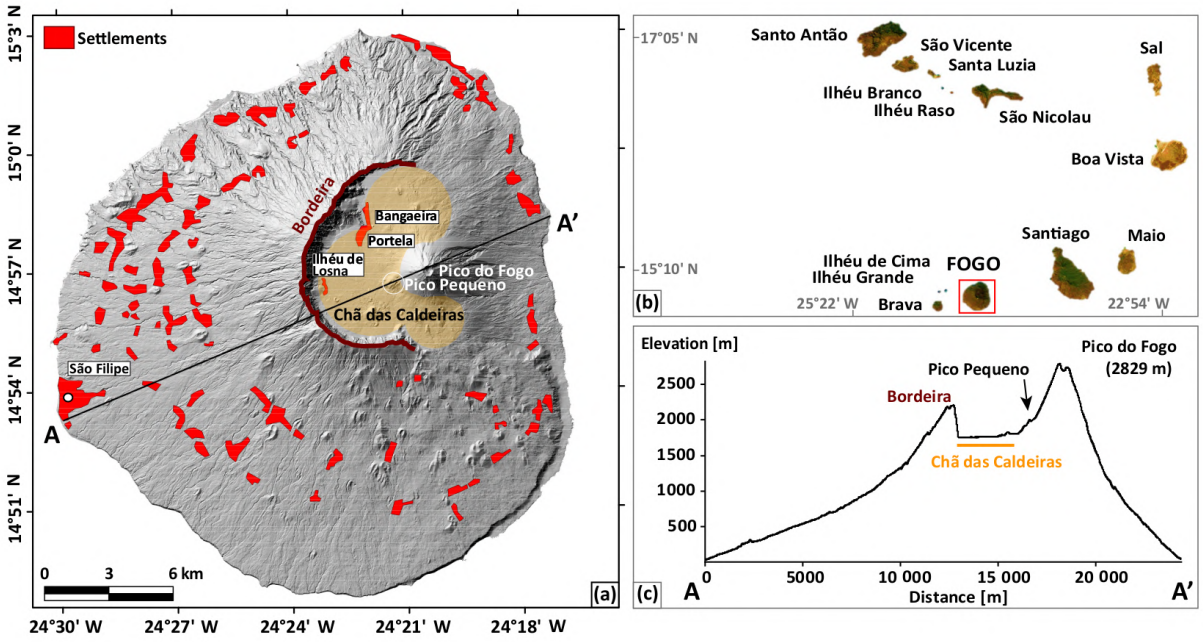


Figure 3.32: (a) Map of Fogo Island. The settlements (red areas) were downloaded from the Copernicus Emergency Management Service (2014), the little settlement of Ilhéu de Losna was added manually to the map. (b) Archipelago of Cape Verde, of which Fogo is a part of. (c) Altimeter profile of the cut A–A' displayed in (a). Images from Richter et al. [231].

Topography and vent location have primary impact in the simulation of real events and all codes, even those that do not consider the rheology and temperature influence on the final flow emplacement have to use these data. Therefore, the more accurate the topography and the vent position, the more reliable the results are. The topography used in our simulations is a DEM (Digital Elevation Model) generated from SAR satellite with data acquired in 2011–2013. It has a horizontal resolution of 12 meter (the DEM of Fogo is a TanDEM-X WorldDEM[1] data [232] provided by the German Aerospace Center (DLR) through data proposal DEM_GEOL_1522, PI Nicole Richter). As stated previously, the eruptive source was a fissure. Richter et al. [231] estimated the lava flow hazard at Fogo by using the probabilistic code DOWNFLOW [83, 252] and used a single vent that corresponds to the highest end of the fissure (DMS coordinates: 14°56′40.56″ N - 24°21′12.28″ W; UTM coordinates: East 784689.69 - North 1653895.03, zone 26N). Cappello et al. [30], in their work for the modeling of lava flow hazard at Fogo, observed, instead, that the other end of the fissure, the lowest, was actually the main source of lava. Having this discordant information, we decided to take into account both the vents, using the following position for the second, DMS coordinates: 14°56′27.15″ N - 24°21′22.96″ W; UTM coordinates: East 784375.00 - North 1653479.0, zone 26N. Cappello et al. [30] used HOTSAT, which is a satellite thermal monitoring system, to retrieve details about the eruption, such as the lava thermal flux and the effusion rate. For the first day of eruption (whose lava flow emplacement is represented in Figure 3.33), they registered a mean effusion rate of 10.5 $\mathrm{m^3\,s^{-1}}$, with peaks between 24–27 $\mathrm{m^3\,s^{-1}}$. In addition, they estimated the extrusion temperature to be 1265°C (1538 K). For our simulations, based on their estimate, we used this temperature coupled with a constant effusion rate equal to the mean value since there is no information about the time variations. Moreover, we adopted their suggested values for density (2600 $\mathrm{kg\,m^{-3}}$) and specific heat capacity (1150 $\mathrm{J\,kg^{-1}\,K^{-1}}$); these values are typical of basaltic magma, which is a proper choice because it respects the characteristics of the magma for Fogo.

For the tests presented here, we adopted a Bingham plastic rheology model with temperature-dependent viscosity, and for this reason, we considered the thermal heat exchanges and solved the system of Eqs. (2.42). We report in Table 3.7 the physical parameters we adopted that do not change in the simulations. The values of the emissivity, exposed area inner fraction, atmospheric heat transfer, and environmental temperature are those suggested by Costa and Macedonio [55] for the eruptions of Etna; since both the lava of Etna and that of Fogo are basaltic, this choice of values is consistent. The value for the thermal conductivity of the soil is a mean value extrapolated from Proceedings World Geothermal Congress 2010 [220]. In the table, we also reported the values characterizing the temperature profile we used for this test case (n, M and T_{soil}).

For the preliminary results concerning the sensitivity analysis to the rheological parameters, we used a spatial discretization grid with $40\,\mathrm{m} \times 40\,\mathrm{m}$ cells, whereas in the next we studied also the effect of different grid resolutions on the results. The PVR (physical variable reconstruction) is used at the interfaces with the generalized minmod limiter (3.6) (with $\theta = 1.3$) and with a 2 stages IMEX R-K scheme for time marching, see Table 3.1. We specify that the simulations computed for this real case test were obtained with a slightly different code with respect to that described above. Differences are related to

[1]The TSX/TanDEM-X mission is for the creation of a global, consistent, and high-resolution Digital Elevation Model (DEM) obtained by exploiting the interferometric capabilities of the two twin SAR satellites TerraSAR-X and TanDEM-X, which fly in a close orbit formation. The work for the creation of this global DEM lasted from December 2010 to September 2016.

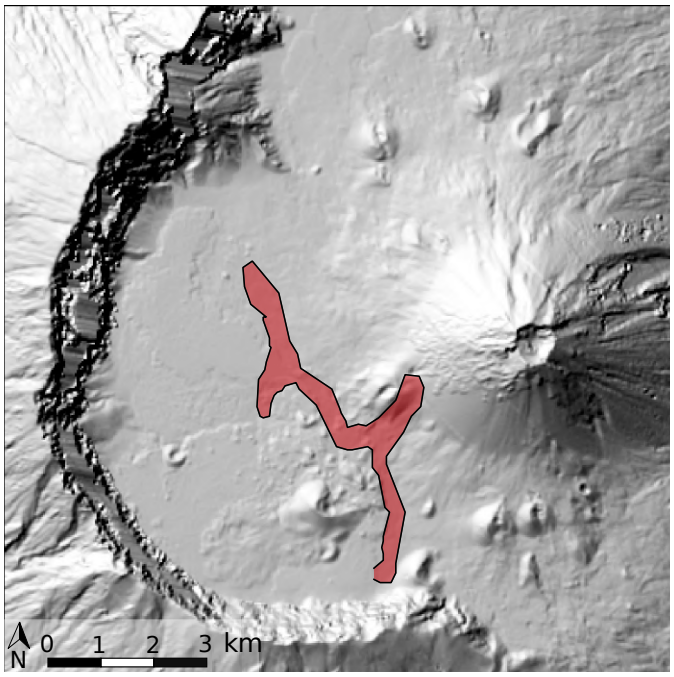


Figure 3.33: Lava flow emplacement of the real event after one day of the eruption, the 24 November 2014. The red area corresponds to the outlines of actual flow fields based on field mapping and satellite images, as reported in [30].

Table 3.7: Parameters of lava flow simulation.

Symbol	Value	Definition	Unit
ρ	2600	density of lava	$\mathrm{kg\,m^{-3}}$
c_p	1150	specific heat of lava	$\mathrm{J\,kg^{-1}\,K^{-1}}$
k_{fl}	4.0	thermal conductivity of lava	$\mathrm{W\,m^{-1}\,K^{-1}}$
k_{soil}	2.0	thermal conductivity of soil	$\mathrm{W\,m^{-1}\,K^{-1}}$
T_{vent}	1538	temp. of lava at the vent	K
T_{env}	300	temp. of environment	K
T_{soil}	300	temp. of soil	K
ϵ	0.8	emissivity of lava	$\mathrm{m^{-2}}$
f	0.5	fract. area of exposed inner core	–
λ	70	atmospheric heat transfer coeff.	$\mathrm{W\,m^{-2}\,K^{-1}}$
n	4	param. of the lava temp. profile	–
M	12	param. of the soil temp. profile	–

optimizations and to the choice of the variables used for the reconstruction at the cell interfaces. In our future works, we will validate this new version of the code.

Sensitivity study to rheological parameters. In this test of lava simulation, since we adopted a Bingham plastic rheology model with a temperature-dependent viscosity, the friction coefficient γ of the momentum equation writes as follows (from Eqs. (2.12) and (2.13)):

$$\gamma = \frac{1}{\rho}\left(\frac{3}{h}\widetilde{\mu} + \frac{\tau_0}{\|\mathbf{U}\|}\right), \qquad \text{with} \qquad \widetilde{\mu} = \mu_{ref}\exp\left[-b\left(T - T_{ref}\right)\right],$$

and the viscous heating term in the temperature equation (see Eq. (2.39)) is

$$\mathcal{K}(U^2 + V^2)e^{-b(T-T_{ref})}.$$

Such model requires to choose the values of the yield stress τ_0, of the parameter b, and of the reference values of temperature and viscosity, T_{ref} and μ_{ref}. We notice that when $b = 0\,\mathrm{K}^{-1}$ the temperature equation is decoupled from the others: the friction coefficient of the momentum equation does not depend on temperature and the same is for the viscous heating term in the temperature equation.

In the following, we show the effects of viscosity, yield stress and temperature on the flow propagation. We assumed that lava dynamic viscosity μ_{ref} is of the order of 10^2–10^4 Pa s upon eruption, namely at the effusion temperature, as suggested in [39, 120, 151, 229] for basaltic magma as the Fogo's. We considered the temperature at the vent as the reference value for the temperature-dependent viscosity model, $T_{ref} = T_{vent} = 1538\,\mathrm{K}$, and produced three sets of simulations respectively with the viscosity at the vent μ_{ref} equal to 100, 1000, and 10000 Pa s. The yield stress τ_0 for basaltic lava is assessed in the range 10^2–10^4 Pa, see Bernabeu et al. [13], so we chose to adopt the three values 100 Pa, 1000 Pa, and 10000 Pa. In addition, we also made simulations for the Newtonian case that consists of $\tau_0 = 0\,\mathrm{Pa}$. The value $2 \times 10^{-2}\,\mathrm{K}^{-1}$ is adopted for the parameter b in Costa and Macedonio [55], in a simulation of Etna volcano. A similar value, $1.6 \times 10^{-2}\,\mathrm{K}^{-1}$, is used in [13], and both situations are relative to basaltic lava. Here, we tested the values $10^{-2}\,\mathrm{K}^{-1}$ and $10^{-3}\,\mathrm{K}^{-1}$ for b and also considered the decoupled case, namely $b = 0\,\mathrm{K}^{-1}$. In the future, we want to compare our results with those descending from the use of the rheological model defined by the VFT equation and discussed, for example, in [102].

Figures 3.34, 3.35, 3.36 show thickness and temperature of lava flow after 24 hours of eruption for three different sets of simulations, with reference dynamic viscosity $\mu_{ref} = 10^2, 10^3, 10^4$ Pa s respectively. In these Figures (as in all the Figures for this test case), we plotted simulation results by applying a 1 cm threshold to the flow thickness to be visualized. The choice of a small threshold for the representations has an impact (however negligible) only on those simulations obtained using small yield stresses, in fact, the higher the yield stress and the thicker the lava front. In the different panels of Figures 3.34, 3.35, 3.36, we present the results obtained by varying yield stress τ_0 and rheological parameter b. Comparing results, we notice that increasing values of the yield stress reflects on narrower and taller lava flow emplacements. For $b = 10^{-2}\,\mathrm{K}^{-1}$ there is the strongest coupling between temperature and velocity, and the temperature has the greatest influence on viscosity. In this case, independently from the other parameters, we observe that the lava flow emplacement is not much affected by the topography and remains close to the vents. The case $\tau_0 = 10^4\,\mathrm{Pa}$ is to consider a little exception, in fact, even though the flow propagation is much limited, it is evident that lava follows the topography. Simulations with $b = 0\,\mathrm{K}^{-1}$ (decoupled equations and constant viscosity) show results far different from those obtained with $b = 10^{-2}\,\mathrm{K}^{-1}$, whereas the case with $b = 10^{-3}\,\mathrm{K}^{-1}$ is similar to the decoupled case ($b = 0\,\mathrm{K}^{-1}$) since the coupling is more weakened with respect to case with $b = 10^{-2}\,\mathrm{K}^{-1}$. The largest emplacement areas of each figure for all the three reference viscosity considered realize in the decoupled Newtonian case ($b = 0\,\mathrm{K}^{-1}, \tau_0 = 0\,\mathrm{Pa}$), because the lower the viscosity the larger the surface area inundated, so that Figure 3.34 presents the largest final emplacement among all simulations.

Our further attempt was to determine ranges for the values of b, μ_{ref}, and τ_0 that produce simulations compatible with the observed event and to study their effects on the runout. By comparing results of Figures 3.34 – 3.36 with the emplacement of the real event

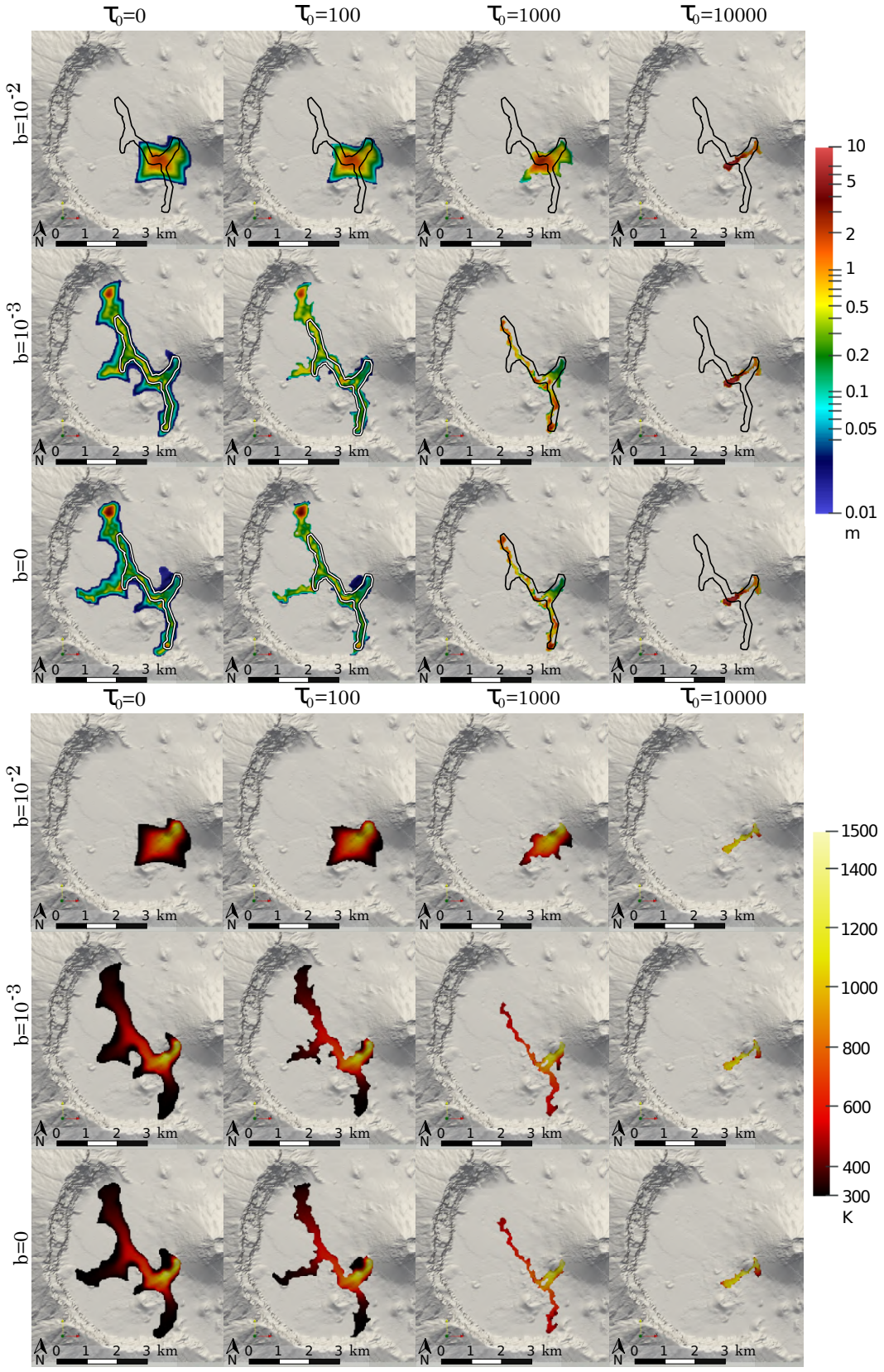


Figure 3.34: Simulations after 24 hours of eruptions computed with a grid size of 40 m, considering $\mu_{ref} = 10^2$ Pa s, varying the yield stress τ_0 (Pa) and the parameter b (K^{-1}). *Top:* the thickness (in logarithmic scale) and the outline of the actual emplacement. *Bottom:* temperature.

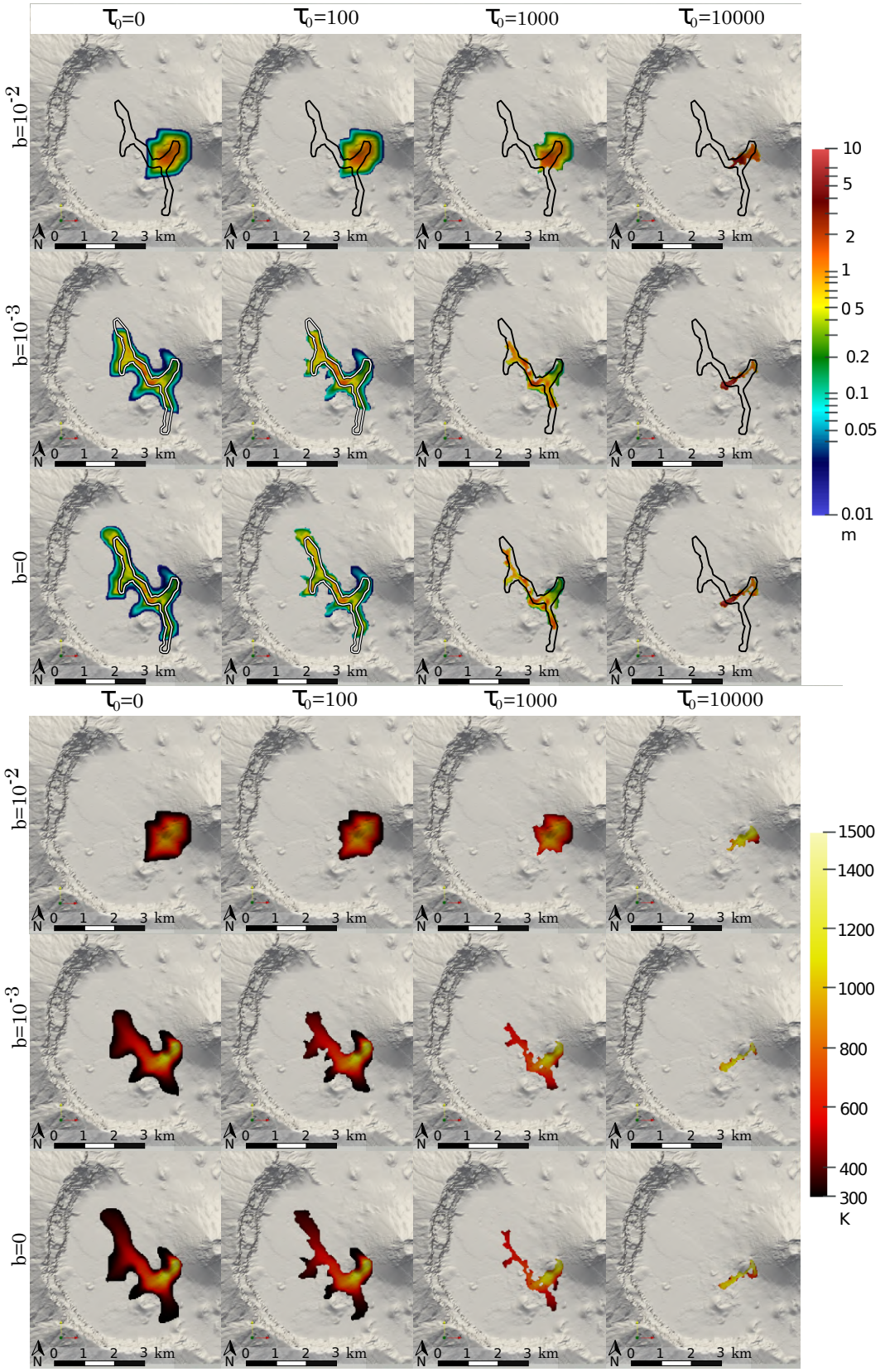


Figure 3.35: Simulations after 24 hours of eruptions computed with a grid size of 40 m, considering $\mu_{ref} = 10^3$ Pa s, varying the yield stress τ_0 (Pa) and the parameter b (K^{-1}). *Top:* thickness (in logarithmic scale) and the outline of the actual emplacement. *Bottom:* temperature.

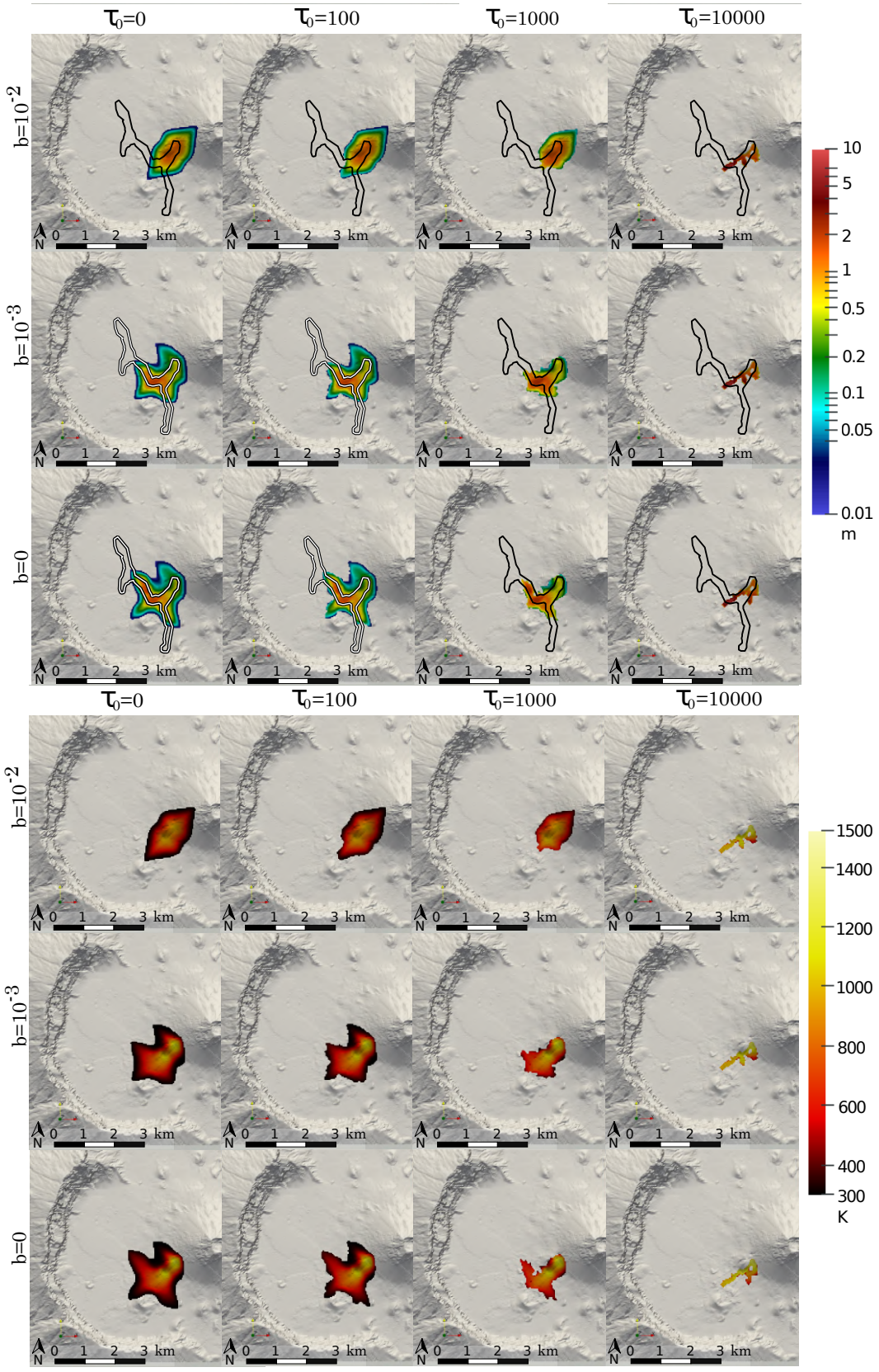


Figure 3.36: Simulations after 24 hours of eruptions computed with a grid size of 40 m, considering $\mu_{ref} = 10^4$ Pa s, varying the yield stress τ_0 (Pa) and the parameter b (K^{-1}). *Top:* thickness (in logarithmic scale) and the outline of the actual emplacement. *Bottom:* temperature.

(see Figure 3.33, Map, and Cappello et al. [30]), we observed that simulations obtained with $b = 10^{-3}\,\mathrm{K}^{-1}$, $10^2 \leqslant \tau_0 \leqslant 10^3\,\mathrm{Pa}$ and $10^2 \leqslant \mu_{ref} \leqslant 10^3\,\mathrm{Pa\,s}$ present more similarities with a real case, so we did a further investigation. By fixing the parameter b, we let τ_0 and μ_{ref} assume some values in the interval $[10^2, 10^3]$ obtaining a more accurate description of the solution dependence on such parameters. Figure 3.37 depicts thickness in logarithmic scale and Figure 3.38 reports temperature. As observed previously, we can confirm that the higher the yield stress and the narrower (and taller on the average) the flow, moreover, the lower the viscosity and the farther the lava propagates. Being viscosity temperature dependent, we also plotted its value in Figure 3.39, with the same reference scale for all the panels (that range between the minimum value of $10^2\,\mathrm{Pa\,s}$, that corresponds to the viscosity at the vent when $\mu_{ref} = 100\,\mathrm{Pa\,s}$, and the maximum value of $3448\,\mathrm{Pa\,s}$, reached when the fluid is at the environment temperature in the case of $\mu_{ref} = 1000\,\mathrm{Pa\,s}$). From this Figure, it is evident (and highlighted by the use of the logarithmic scale) that viscosity values strongly depend on the reference viscosity considered. Hence, we passed to consider, for each reference viscosity, the respective range of temperature-dependent viscosity and plotted the results in Figure 3.40 where the maximum values scale in relation to the reference value of viscosity.

Comparing the results shown in Figures 3.37 and 3.38 with the real lava flow emplacement after one day of eruption in Figure 3.33, we noticed that the simulation obtained with $\mu_{ref} = 100\,\mathrm{Pa\,s}$ and $\tau_0 = 500, 750\,\mathrm{Pa}$ are those more similar to the real event. Since more than one simulation shows a good agreement with the observations, we fix one choice of parameters to continue with further analysis: namely $\mu_{ref} = 100\,\mathrm{Pa\,s}$ and $\tau_0 = 750\,\mathrm{Pa}$. In future works, we plan to evaluate the quality of fit between each simulation and the real emplacement by using the measurements defined in [52] based on the area of overlap, over-extent, and under-extent between the footprints of the observed and simulated lava.

Figure 3.41 shows the story of the first 24 hours of eruption with the previous choice of parameters: $b = 10^{-3}\,\mathrm{K}^{-1}$, $\mu_{ref} = 100\,\mathrm{Pa\,s}$ and $\tau_0 = 750\,\mathrm{Pa}$. The main north and south branches developed meanwhile (fact that happened also in the real eruption) and the propagation velocity of them was similar (at least for the first 24 hours). After 12 hours of eruption, the north lobe extended for $\sim 1200\,\mathrm{m}$ and the south lobe for $\sim 800\,\mathrm{m}$. A third lobe developed after 15 hours of eruption west going from the north branch and reached a maximum length of $\sim 500\,\mathrm{m}$. The north-lobe extended till a maximum length of $\sim 2400\,\mathrm{m}$ in 24 hours, instead the south-lobe reached Bordeira after 21 hours and a final extension of $\sim 1800\,\mathrm{m}$

As previously stated, in the simulations presented so far, we assumed the flow is feeded by two vents to compute our simulations. In order to analyze the effect of this assumption, in Figure 3.42 we compare the results of simulations obtained considering both the vents or only one of them. Even if the final lava flow emplacements are not so different, we can appreciate that in the cases where only one vent is adopted the final front propagates more towards north or south alternatively, showing that a proper choice of the vent location (and its changes during an eruption) are important for an accurate description of lava flow emplacement.

We also tested the sensitivity of numerical results to the grid resolution by comparing simulations obtained with three different cell sizes (see Figure 3.43): coarse with $80\,\mathrm{m} \times 80\,\mathrm{m}$ cells, medium with $40\,\mathrm{m} \times 40\,\mathrm{m}$ cells, and fine with $20\,\mathrm{m} \times 20\,\mathrm{m}$ cells. Result obtained with the coarse mesh is far from what obtained with the fine mesh, instead the runout computed with the medium mesh is comparable to that, therefore we can conclude that simulations produced using a mesh $40\,\mathrm{m} \times 40\,\mathrm{m}$ are quite reliable. Table 3.8 show the

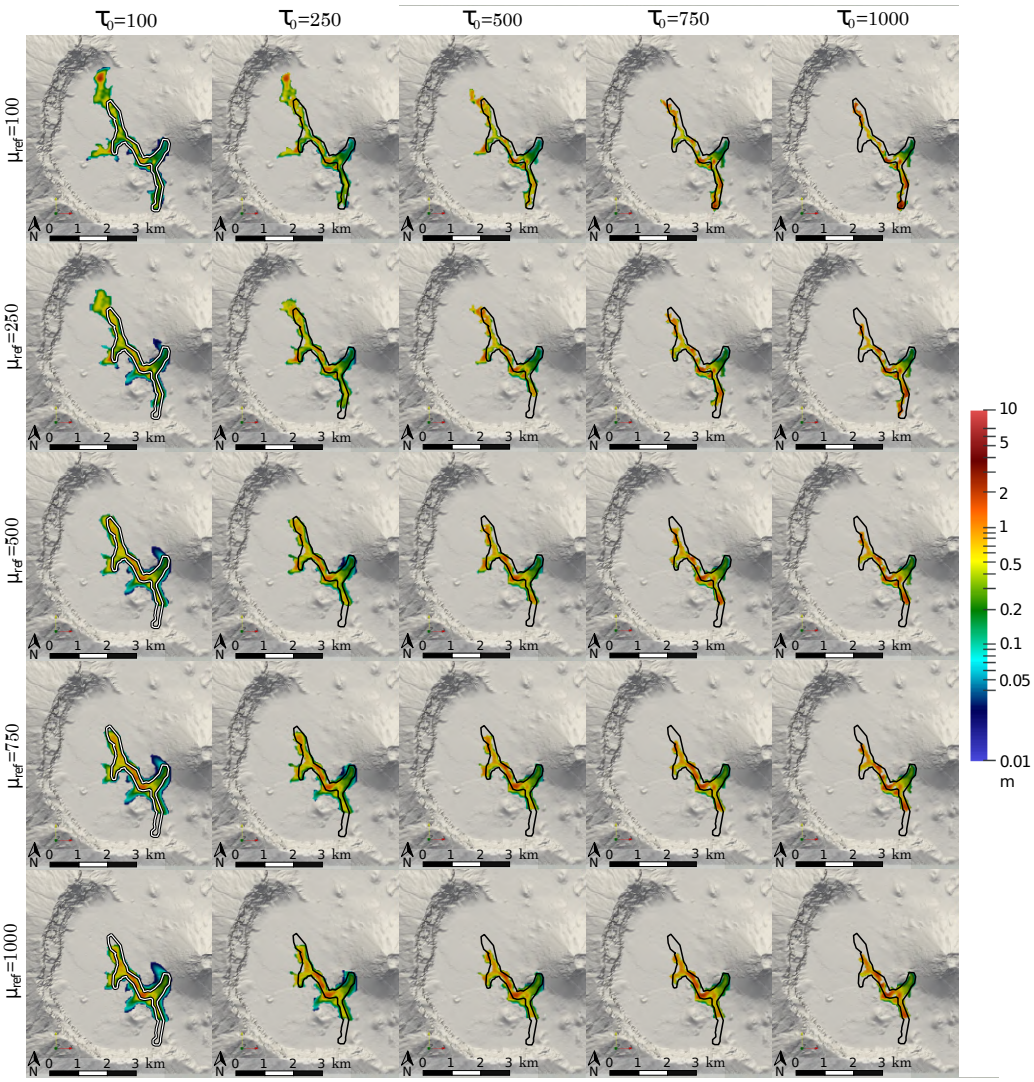


Figure 3.37: *Thickness (in logarithmic scale).* Simulations after 24 hours of eruptions computed with a grid size of 40 m, fixing $b = 10^{-3}\,\mathrm{K}^{-1}$ and varying both the yield stress τ_0 and the reference viscosity μ_{ref} in the interval $[100, 1000]$. The outline represents the actual lava flow emplacement.

execution time of the three simulations computed with the processor Intel® Core™ i7-6500U CPU, 2.50 GHz×4. The short execution times allow using our model also for hazard quantification and for the production of probabilistic maps because many simulations can be performed in a short time.

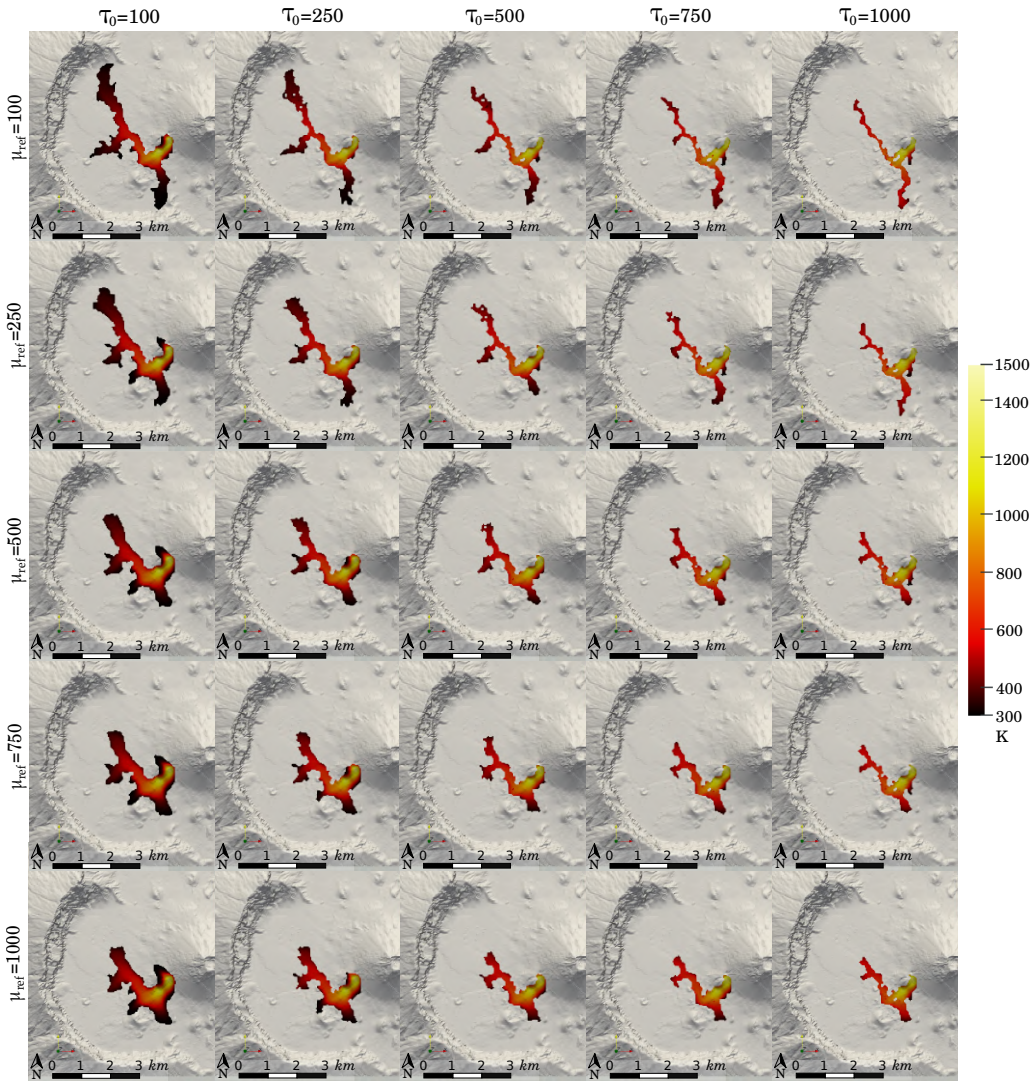


Figure 3.38: *Temperature.* Simulations after 24 hours of eruptions computed with a grid size of 40 m, fixing $b = 10^{-3}\,\mathrm{K}^{-1}$ and varying both the yield stress τ_0 and the reference viscosity μ_{ref} in the interval $[100, 1000]$.

cells	80×80 m	40×40 m	20×20 m
time	100 s	539 s	2606 s

Table 3.8: Elapsed time for the execution of simulations with different grid sizes; simulations of 24 hours of eruptions computed using reference viscosity $\mu_{ref} = 100$ Pa s, rheological parameter $b = 10^{-3}\,\mathrm{K}^{-1}$, and yield stress $\tau_0 = 750$ Pa. Processor specifications: Intel® Core™ i7-6500U CPU, 2.50 GHz × 4.

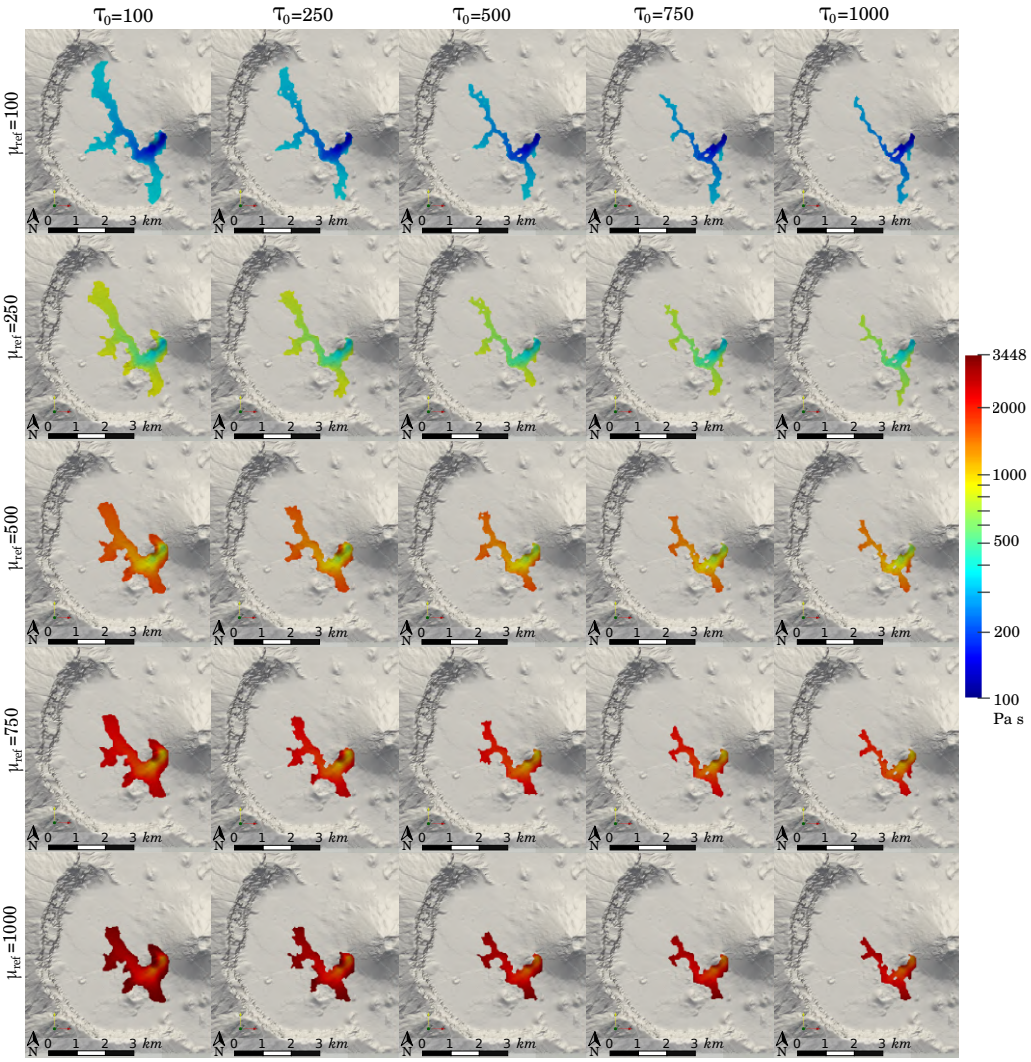


Figure 3.39: *Temperature dependent viscosity (unique logarithmic scale).* Simulations after 24 hours of eruptions computed with a grid size of 40 m, fixing $b = 10^{-3}$ and varying both the yield stress τ_0 and the reference viscosity μ_{ref} in the interval $[100, 1000]$.

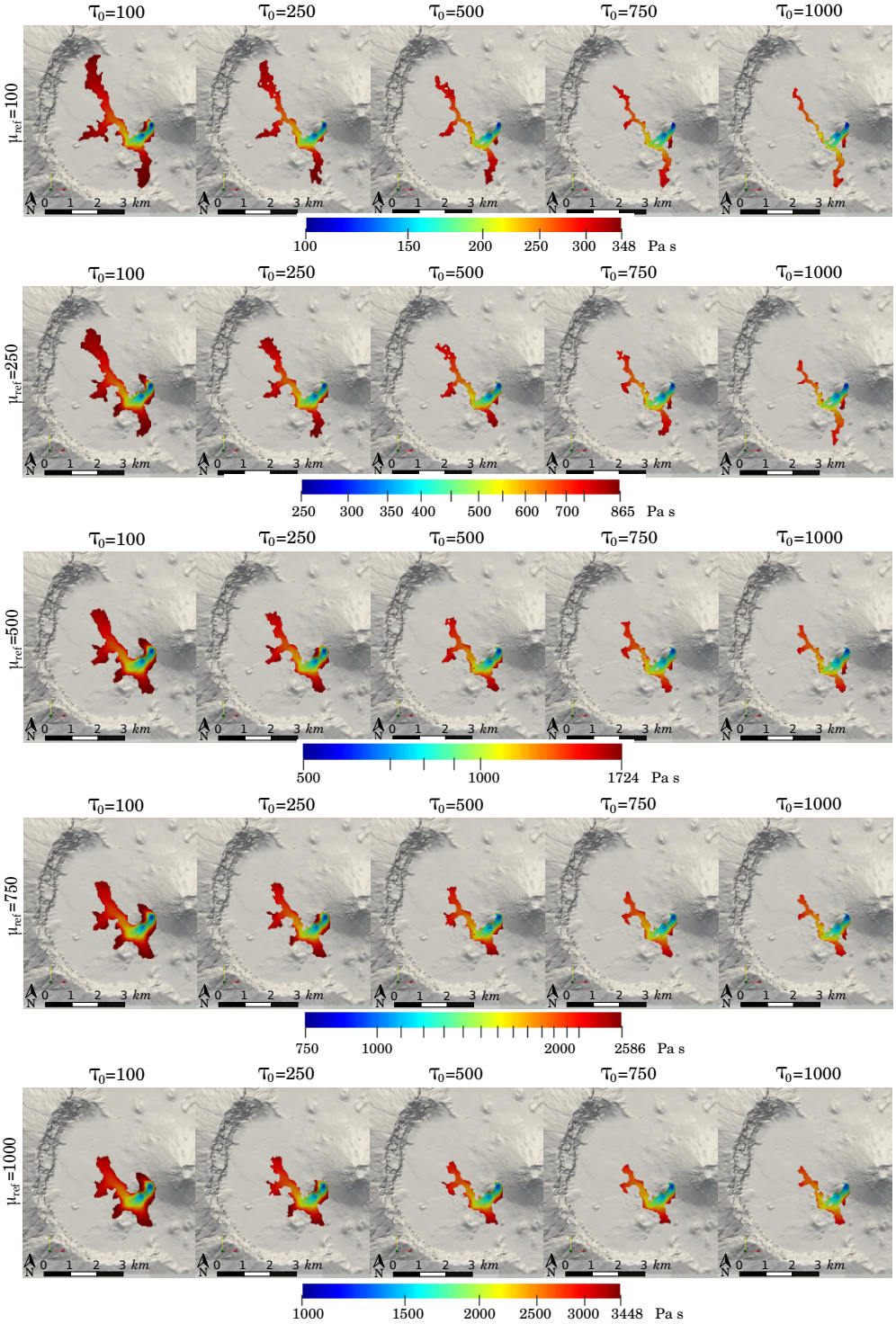


Figure 3.40: *Temperature-dependent viscosity (each logarithmic scale depends on the reference viscosity).* Simulations after 24 hours of eruptions computed with a grid size of 40 m, fixing $b = 10^{-3}\,\mathrm{K}^{-1}$ and varying both the yield stress τ_0 and the reference viscosity μ_{ref} in the interval $[100, 1000]$.

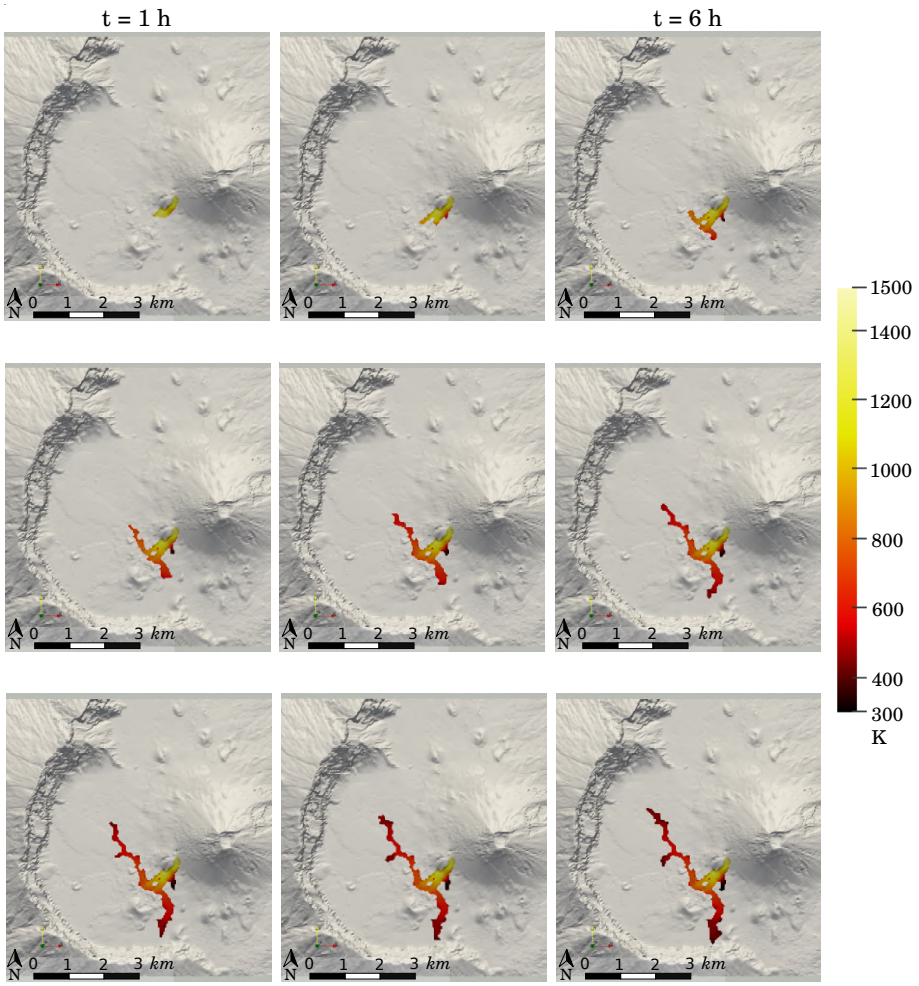


Figure 3.41: *History simulation of one day of eruption.* Grid size of 40 m, reference viscosity $\mu_{ref} = 100$ Pa s, rheological parameter $b = 10^{-3}\,\mathrm{K}^{-1}$, and yield stress $\tau_0 = 750$ Pa.

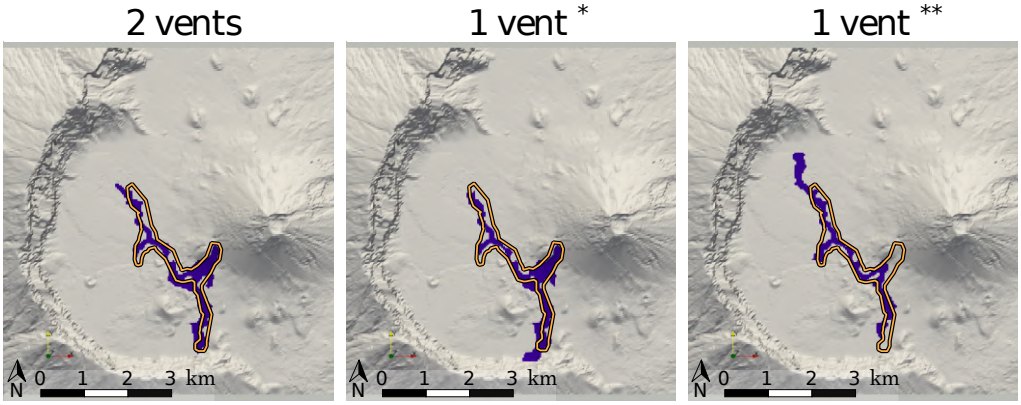


Figure 3.42: *Sensitivity to the vent position.* We call V1 and V2 the two vents we employed in our simulations: V1 (DMS coordinates: 14°56′40.56″ N - 24°21′12.28″ W; UTM coordinates: East 784689.69 - North 1653895.03, zone 26N), and V2 (DMS coordinates: 14°56′27.15″ N - 24°21′22.96″ W; UTM coordinates: East 784375.00 - North 1653479.0, zone 26N). *Left:* both vents were adopted. *Center:* we used V1, the highest vent. *Right:* the lower vent V2 was adopted. Simulations after 24 hours of eruptions computed with a grid size of 40 m, reference viscosity $\mu_{ref} = 100$ Pa s, rheological parameter $b = 10^{-3}\,\mathrm{K}^{-1}$, and yield stress $\tau_0 = 750$ Pa. The outline represents the actual lava flow emplacement.

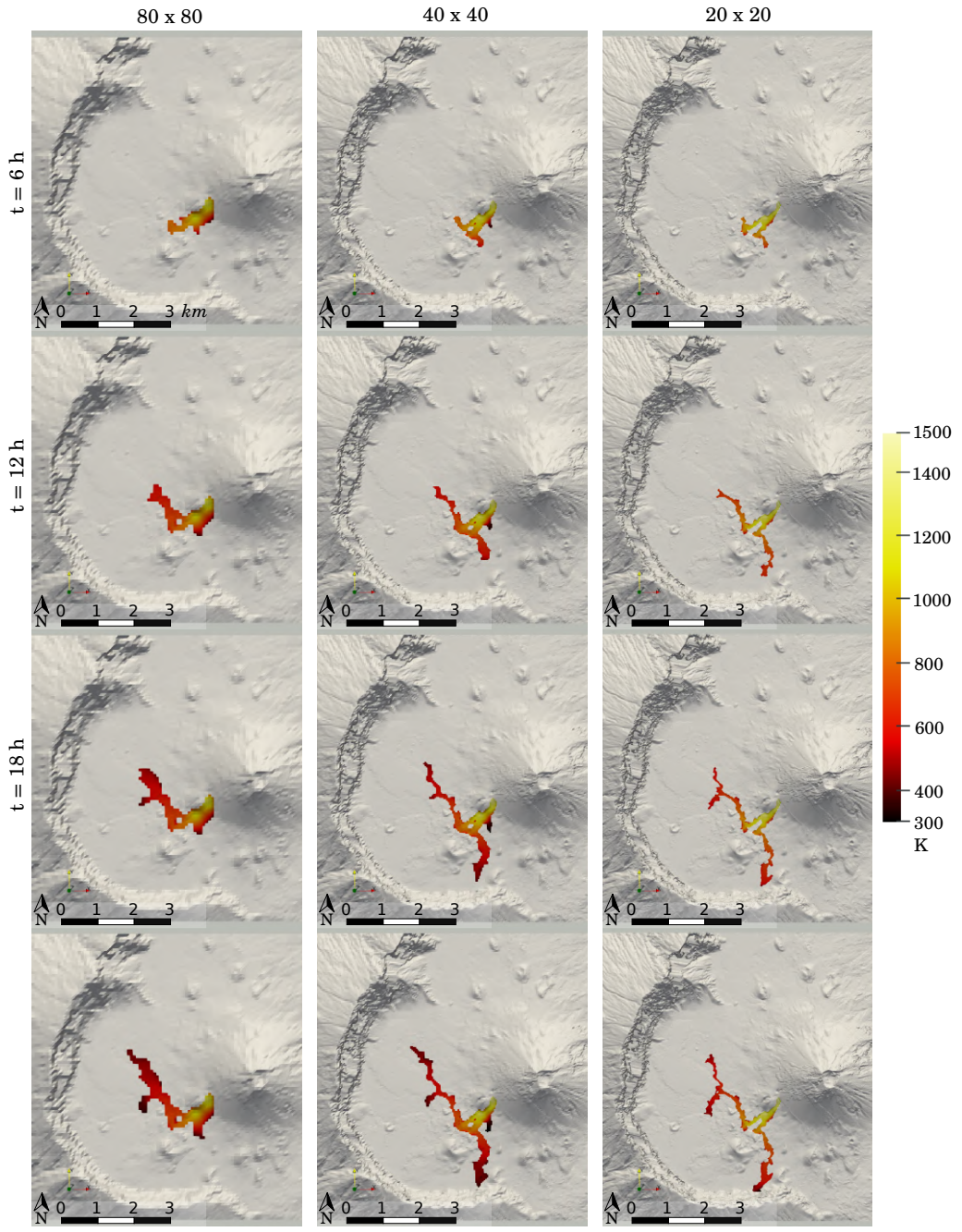


Figure 3.43: *Sensitivity to the numerical grid size. Left:* grid size $80 \times 80\,\mathrm{m}$. *Center:* grid size $40 \times 40\,\mathrm{m}$. *Right:* grid size $20 \times 20\,\mathrm{m}$. Simulations of 24 hours of eruptions computed with reference viscosity $\mu_{ref} = 100\,\mathrm{Pa\,s}$, rheological parameter $b = 10^{-3}\,\mathrm{K}^{-1}$, and yield stress $\tau_0 = 750$ Pa.

Chapter 4

Numerical discretization of the 3D model

3D models permit an accurate description of fluid dynamics processes, in the first place, and of thermophysical and rheological processes, as a consequence. Vertical distribution of the variables (as velocity, temperature, and viscosity) is directly treated. In the context of the description of the free-surface fluids, most 3D models result in a multiphase model (accounting also the overlying air) as ours, and this feature produces a further increase in the numerical and computational efforts necessary for solving the governing equations. The sum of these difficulties slowed the development and popularity of 3D models in the past. Thanks to the technology development and increase in the computational power that happened in the latest decades, such as the possibility of parallelizing the code execution, CFD software for 3D models evolved and spread. Engineers and researchers commonly use tools for 3D CFD modeling, which range from commercial software packages to community-driven or government-supported open-source libraries. We quote below four advanced CFD tools, three commercial and one open-source, that can be used for lava flow simulations. The application of such tools to lava flow simulations may require the implementation of additional capabilities that might not be built-in, like the rheology and thermodynamics models and the possibility to employ topography.

ANSYS® FLUENT is a CFD commercial software oriented to engineering use that implements 3D models and adopts a cell-centered numerical approach to the FVM. That software makes use of pre- and post-processing and also visualization programs. The codes are parallelized with MPI standard. The company of engineering simulation ANSYS® takes care of the software maintenance. Such a company was founded in 1970 in Canonsburg, Pennsylvania, and FLUENT is only one of the numerous products they develop.

FLOW-3D® is a commercial software devoted primarily to engineering with a particular focus on 3D CFD. Multiphase modeling bases on the VOF method applied in combination with the level-set formulation (both belong to the family of interface-capturing methods we have seen in §2.2). This software, used for free-surface flows, can model all types of heat transfer, flows on porous media, and viscous fluids. Hence, FLOW-3D lends itself to be also used for 3D lava flow modeling. The code parallelization relies on the OpenMP paradigm. FLOW-3D is produced and distributed by Flow Science Inc. and is a popular choice for a wide range of industrial applications. The two main disadvantages of FLOW-3D are the slowness and the high price. In return, users get a fully developed and tested modeling environment with a graphical user interface and product support.

COMSOL® Multiphysics is a simulation commercial software, developed by COMSOL Inc., that provides an IDE and unified workflow for electrical, mechanical, fluid, acoustics, and chemical applications, which is why it is referred to as "Multiphysics". The numerical base-core of COMSOL is the finite element method (FEM).

OpenFOAM [137] is an open-source software package produced by OpenCFD Ltd. devoted mainly to CFD (we already introduced OpenFOAM in §1.3; here, we remind its main features and compare it to the other software for 3D modeling to motivate our decision of using it for our work). The OpenFOAM solvers are based on FVM schemes and can deal with complex fluids, chemical reactions, turbulence, heat transfer, solid mechanics, and electromagnetics. Codes are parallelized with OpenMPI, and the software uses pre- and post-processing tools. Being open-source software, users can modify solvers and applications present besides using them directly as they are. So, even though OpenFOAM does not have a solver that precisely models lava flows with their complex physics, the user can create that according to his needs.

We decided to develop our work in the OpenFOAM framework because of its modeling features and open-source nature. In particular, we identified `interFoam` solver as the starting point of our work. `interFoam` solves the dynamics for two incompressible, isothermal, and immiscible fluids using the Volume Of Fluid method, and its performances are analyzed in Deshpande et al. [69]. Our work consisted of modifying such a solver by adding the energy equation that accounted for diffusion and the radiative and convective heat exchanges with the environment. Heat conduction with soil is instead implemented as a boundary condition. In our new solver that is called `interThermalRadConvFoam`, we also implemented a temperature-dependent viscosity, the possibility to import a DEM (Digital Elevation Model) file (in order to compute simulation over real topography), the dynamic mesh refinement, and the parallelization.

We recall here the system of PDEs we derived in §2.2 to describe the dynamics of two immiscible multiphase fluids with Newtonian viscosity, in incompressible flow condition, that accounts also the evolution of the thermal energy caused by heat exchanges:

$$\nabla \cdot \mathbf{u} = 0 \tag{4.1a}$$

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot \left(\rho \mathbf{u}\mathbf{u}^T\right) = -\nabla p + \nabla \cdot \boldsymbol{\tau} + \rho \mathbf{g} + \mathbf{f}_\Sigma \tag{4.1b}$$

$$\partial_t \alpha + \nabla \cdot (\mathbf{u}\alpha) = 0, \tag{4.1c}$$

$$\frac{\partial(\rho c_p T)}{\partial t} + \nabla \cdot (\rho c_p T \mathbf{u}) - \bar{\chi}_\Sigma \Delta(kT) = -\frac{\epsilon \sigma_{SB} f A_{fs}}{Vol}\left(T^4 - T_{env}^4\right) - \frac{\lambda f A_{fs}}{Vol}\left(T - T_{env}\right). \tag{4.1d}$$

Eq. (4.1a) refers to the mass conservation principle and consists in a kinematic constraint on the velocity field $\mathbf{u}$ (because of the incompressible flow assumption, see §1.1.5.6). Eq. (4.1b) is the momentum conservation equation, and its non hyperbolic terms are, respectively: the opposite of the gradient of the pressure field p, the divergence of the viscous stress tensor $\boldsymbol{\tau}$ (defined in Eq. (1.47)), the gravity force with the gravity acceleration g, and finally the surface tension $\mathbf{f}_\Sigma$ (defined in Eq. (2.57)). Eq. (4.1c) is the transport equation for the liquid volume fraction α, the variable that takes the trace of the two fluids described, derived in §2.2.1.1. Eq. (4.1d) for energy (derived in §2.2.2) has the hyperbolic and diffusive terms on the left hand side, and the terms for radiative and convective heat exchanges on the right. T is the temperature variable, instead the remaining parameters that appear in the equation are: c_p the specific heat, k the thermal conductivity, $\bar{\chi}_\Sigma(\mathbf{x}, t)$ a function that suppresses the thermal diffusion between the fluids,

ε the emissivity, σ_{SB} the Stefan-Boltzmann constant, f the fractional area of the exposed inner core, T_{env} the environmental temperature, λ the heat transfer coefficient. Since the radiative and convective phenomena happen on the fluid free surface, the two terms describing them activate only in correspondence of that and so are both proportional to the A_{fs} that is the area of the free surface.

The unknowns of the system of Eqs. (4.1) are velocity $\mathbf{u}$, pressure p, temperature T, and the liquid-phase volumetric fraction α, whereas density ρ is assumed constant (in each phase). From a numerical point of view, the system presents several problems and frailties:

- The continuity equation is expressed by a kinematic constraint; consequently, an explicit condition for pressure is missing. Such a problem is overcome by adopting a scheme that belongs to the segregated methods family, exposed in §4.1, that consists of deriving and solving a Poisson equation for pressure.

- The momentum equation is non-linear in the advective term, hence in the cases of implicit treatment, some sort of linearization becomes fundamental, as shown in §4.1.

- At the initial time, the indicator function of the liquid-phase volumetric fraction presents a discontinuity at the interface between the two phases and assumes values zero or one. Similarly, the numerical schemes employed to solve the transport equation for α must preserve the values of α bounded, between zero and one, and at the same time must maintain the interface as sharp as possible. For these reasons, the choice of the scheme for the advective term discretization is of paramount importance. Moreover, since adopting a fine computational grid helps to describe the interface accurately, one could think of playing with the numerical grid resolution. A common method used to keep a sharp interface is a sub-grid level reconstruction of the interface using linear or quadratic polynomials. However, the choice of a good grid resolution does not guarantee boundedness of α, nor sharpness of the interface through time. Furthermore, the use of a finer grid has the drawback to increases a lot the computational cost. The method implemented in `interFoam` is different, it follows the Flux Corrected Transport (FCT) approach described by Boris and Book [22] adopting the Multidimensional Universal Limiter for Explicit Solution (MULES) scheme depicted by Márquez Damián [196], and we will present such a scheme in §4.3 and §4.4.

In addition, we underline that our solver `interThermalRadConvFoam` (but also the original `interFoam`) does not solve exactly the Eqs. (4.1), but a slightly modified version. One difference is in the transport equation for α and consists in the introduction of an artificial term useful to maintain the interface between the fluids sharper, described in §4.2. The other difference is in the right-hand side terms of the momentum equation, which sees a rearranging of the pressure and gravitational terms, as discussed in detail in §4.5.

The outline of the Chapter is as follows: the segregated approach employed to solve the mass-momentum system of equations is described in §4.1; in §4.2 we describe FCT discretization technique adopted for the discretization of the α equation; in §4.3 and §4.4 the MULES scheme is described; the design of our solver `interThermalRadConvFoam` is presented in §4.5; in §4.6 we give a brief sketch of numerical approaches different from the segregated approach adopted here, and, finally, in §4.7 we presents the results of our simulations.

4.1 Segregated Method

To solve a system of coupled PDEs, like the Navier-Stokes Eqs. (1.37) numerically (or even the incompressible case of Eqs. (1.49) or the system of our incompressible multiphase model of Eqs. (4.1)), there are two possible ways. One option is to follow the ***fully coupled*** approach: all the equations are discretized and linearized together by producing a single large linear system (where the unknowns are all the flow variables), which is solved as a whole (usually through iterative methods for linear systems). The other possibility to solve the system is to adopt the ***segregated*** approach as most of the OpenFOAM solvers do: the equations are solved one at time and the coupling is obtained by an *iterative procedure* (see the original works of Harlow and Welsh [115], Chorin [41], Patankar and Spadling [208], Caretto et al. [33], Issa [134]). In this thesis, we adopt the latter choice to solve our incompressible and multiphase model of Eqs. (2.59).

From a mathematical point of view, this approach allows using different solution methods by choosing those that better suit the mathematical feature of each equation. The most important thing to understand in the segregated algorithms is the physical motivations behind each mathematical coupling between the equations because such motivations drive the construction of the coupling algorithms.

In order to introduce the strategies to couple the equations of our system of Eqs. (4.1), we start by describing the simpler case of single-phase dynamics in which just the mass and momentum equations are considered. The algorithm that is adopted to solve the entire system of Eqs. (2.59), which will be exposed in §4.5, is based on a similar approach, where the solution of additional equations is nested in the procedure. So, the equations for the mass and momentum conservation are recalled here:

$$\nabla \cdot \mathbf{u} = 0, \tag{4.2a}$$

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla \cdot (\rho\mathbf{u}\mathbf{u}^T) = -\nabla p + \nabla \cdot \boldsymbol{\tau} + \rho\mathbf{g}. \tag{4.2b}$$

Since density is constant, then the unknown fields are velocity $\mathbf{u}$ and pressure p. With respect to the compressible formulation, the continuity equation does not have a transient term and it reduces to a kinematic constraint on the velocity. We recall also that, being the flow incompressible (and also the density constant), there is no any state equation linking the pressure with density and p is the mechanical pressure. Because of that, the absolute value of pressure is of no significance and only its gradient is necessary (in the momentum equations). In some sense, the pressure in the incompressible Navier-Stokes equations is entirely mathematical, thus an expression for it should be derived from mathematical relationships. By taking the divergence of the momentum equation and by imposing the kinematic condition of the continuity equation, a Poisson equation for pressure is derived, as shown in the following. It is important to underline that, even in the derived Poisson equation, only the gradient of pressure is involved and not the absolute value of pressure. The discretization of the derivatives in the Poisson equation requires particular attention in order to guarantee consistency with the discretizations applied in the momentum equation, as is stressed in the next paragraphs.

The numerical algorithm to solve the system of PDEs (4.2) starts by moving the advective term of the momentum Eq. (4.2b) on the right-hand side, then a spatial semi-discretization is first applied to the equation. Moreover, we highlight that different schemes may be used for each term. The choice of the numerical discretization scheme of the spatial derivatives is not of interest here, so the generic symbol $\delta/\delta x$ is used to

represent such arbitrary schemes. The momentum equation for the i-th component of the velocity in each cell of the computational grid is represented as

$$\frac{\partial(\rho u_i)}{\partial t} = -\sum_{j=1}^{3} \frac{\delta(\rho u_i u_j)}{\delta x_j} - \frac{\delta p}{\delta x_i} + \sum_{j=1}^{3} \frac{\delta \tau_{ij}}{\delta x_j} + g_i, \quad i = 1, \ldots, 3;$$

here $\delta/\delta x$ may be a different kind of spatial approximation for each term. Furthermore, we recall that, being $\mathcal{N}$ the number of cells that constitute the discretized domain, the total number of equations to solve is proportional to $3\mathcal{N}$, and that any equation involves also contiguous cells because of the discretization schemes of the differential terms (see what described in §1.3.2.1).

The right-hand side terms, with the exception of the pressure term, are collected together in the H-operator:

$$H_i := -\sum_{j=1}^{3} \frac{\delta(\rho u_i u_j)}{\delta x_j} + \sum_{j=1}^{3} \frac{\delta \tau_{ij}}{\delta x_j} + g_i, \quad i = 1, \ldots, 3, \tag{4.3}$$

which acts also as a shorthand notation since here there is no interest in the numerical treatment of the corresponding terms. With this notation, the semi-discretized form of the momentum equation for the i-th component of the velocity is

$$\frac{\partial(\rho u_i)}{\partial t} = H_i - \frac{\delta p}{\delta x_i}, \quad i = 1, \ldots, 3. \tag{4.4}$$

For a full-discretized form of the momentum equation, we also need a discretization in time. To this aim, we first introduce the superscript notation n to denote the numerical approximation of the variables at the time step $t_n = n\Delta t$.

A fast overview is given to the pro and cons that meet in treating in an explicit or implicit way the two source terms of Eq. (4.4), the H-operator and the pressure gradient. Also, the strategies that try to overcome the possible problems are exposed. This analysis provides the basic ideas that are implemented and then used in an iterative fashion in the popular SIMPLE and PISO algorithms, as we see in the following.

Case A. When we apply the first-order forward Euler scheme for the time discretization of the Eq. (4.4), we obtain the following full-discretized form of the momentum equation for the i-th component of the velocity:

$$\frac{\rho u_i^{n+1} - \rho u_i^n}{\Delta t} = H_i^n - \frac{\delta p^n}{\delta x_i}, \quad i = 1, \ldots, 3,$$

where the velocity at time n is used in the computation of H_i^n. Once a numerical scheme for the computation of the pressure gradient $\delta p^n/\delta x_i$ is chosen, it is possible to determine u_i^{n+1} at the new time step. Generally, the velocity field computed in this way does not satisfy the continuity equation, namely it does not guarantee the divergence-free condition, and thus it does not satisfy the mass conservation. It results that, to overcome this problem and enforce this constrain, the velocity and pressure fields used on the right-hand term cannot be both treated explicitly.

Case B. Let consider now an implicit discretization of the pressure term, while retaining the explicit discretization of the operator H.

$$\frac{\rho u_i^{n+1} - \rho u_i^n}{\Delta t} = H_i^n - \frac{\delta p^{n+1}}{\delta x_i}, \quad i = 1, \ldots, 3, \tag{4.5}$$

In this case, if we take the numerical divergence of the full-discretized momentum equation (4.5), we have

$$\sum_{i=1}^{3} \frac{\delta u_i^{n+1}}{\delta x_i} - \sum_{i=1}^{3} \frac{\delta u_i^n}{\delta x_i} = \frac{\Delta t}{\rho} \left[\sum_{i=1}^{3} \frac{\delta}{\delta x_i} \left(H_i^n - \frac{\delta p^{n+1}}{\delta x_i} \right) \right]. \tag{4.6}$$

We observe that, because the continuity is respected at the time step t_n, the second term on the left-hand side is zero. If also the term in square brackets is null, then it follows that even the first term on the left-hand side will be null, resulting in the divergence-free condition at the new time step. By considering this, we have derived the desired condition, which is called the Poisson equation for pressure, written in the discretized form as

$$\sum_{i=1}^{3} \frac{\delta}{\delta x_i} \left(\frac{\delta p^{n+1}}{\delta x_i} \right) = \sum_{i=1}^{3} \frac{\delta H_i^n}{\delta x_i} \tag{4.7}$$

where the outer differential operators on both sides correspond to the numerical divergence just applied and hence they must be discretized in the same way (that would be the same as that used for the numerical discretization of the continuity equation (4.2a)). The pressure p^{n+1} that satisfies the Poisson equation makes the velocity field $\mathbf{u}^{n+1}$ divergence-free in terms of the discrete divergence operator.

The algorithm for time-advancing the Navier-Stokes equations descending from this scheme is:

- Start at time t_n with the velocity field $\mathbf{u}^n$ that is assumed to be divergence-free.
- Assemble the term H_i^n (defined in Eq. (4.3)) and compute its numerical divergence

$$\sum_{i=1}^{3} \delta \frac{H_i^n}{\delta x_i}.$$

- Solve the discrete Poisson equation (4.7) to determine p^{n+1}.
- Compute the divergence-free velocity field $\mathbf{u}^{n+1}$ from the momentum equation at the new time step t_{n+1}:

$$u_i^{n+1} = u_i^n + \frac{\Delta t}{\rho} \left(H_i^n - \frac{\delta p^{n+1}}{\delta x_i} \right), \quad i = 1, \ldots, 3.$$

- Pass to the next time step.

Observe that the final pressure and velocity fields, p^{n+1} and $\mathbf{u}_i^{n+1}$, respect both the mass and momentum equations (4.5). However, this scheme has the disadvantage that the H-operator is evaluated explicitly imposing severe limitations on the maximum time-step guaranteeing the stability of the numerical scheme.

Case C. Let adopt, instead, an explicit discretization for the pressure term and an implicit discretization for the H-operator (defined in Eq. (4.3)) in the momentum equation (4.4). The full-discretized form of the momentum equation for the i-th component of velocity is written as follows

$$\frac{\rho u_i^{n+1} - \rho u_i^n}{\Delta t} = H_i^{n+1} - \frac{\delta p^n}{\delta x_i}, \quad i = 1, \dots, 3, \tag{4.8}$$

where the velocity at the new time t_{n+1} is used in the computation of H_i^{n+1}. Since the unknown is only the velocity field $\mathbf{u}^{n+1}$, it is possible to find the solution. However, the implicit discretization of the H-operator introduces another problem: the H-operator is quadratic with respect to the unknown velocity field because it contains the discretized advective terms (see the definition in Eq. (4.3)). As a consequence, the H-operator must be linearized (namely the advective term is linearized) and the Eq. (4.8) must be solved with an iterative procedure. To linearize the advective term inside H_i^{n+1}, we consider the advected quantity, namely the momentum, as the unknown at the new time step and we retain at the previous time step the advective velocity:

$$\sum_{j=1}^{3} \frac{\delta(\rho u_j^{n+1} u_i^{n+1})}{\delta x_j} \quad \xrightarrow{\text{(lineariz.)}} \quad \sum_{j=1}^{3} \frac{\delta\left((\rho u_j^n) u_i^{n+1}\right)}{\delta x_j}, \quad i = 1, \dots, 3 \tag{4.9}$$

In addition, we observe that also in this case, as for **Case A**, the new velocity field computed in this way may not satisfy the condition of null divergence.

Case D. We consider, for the last case, a fully implicit treatment. The full-discretized form of the momentum equation we have to solve is

$$\frac{\rho u_i^{n+1} - \rho u_i^n}{\Delta t} = H_i^{n+1} - \frac{\delta p^{n+1}}{\delta x_i}, \quad i = 1, \dots, 3. \tag{4.10}$$

We get a system of three equations in the four unknowns of pressure and three velocity components, hence we need an additional constraint to fully determine the solution, which is given by the divergence-free condition. To solve this system of equations, we first try to repeat the same steps done in **Case B** analyzed above. By taking the numerical divergence of the momentum equation (4.10), we obtain something similar to Eq. (4.6), but with the H-operator computed at the time $n+1$ in place of H_i^n:

$$\sum_{i=1}^{3} \frac{\delta u_i^{n+1}}{\delta x_i} - \sum_{i=1}^{3} \frac{\delta u_i^n}{\delta x_i} = \frac{\Delta t}{\rho} \left[\sum_{i=1}^{3} \frac{\delta}{\delta x_i} \left(H_i^{n+1} - \frac{\delta p^{n+1}}{\delta x_i} \right) \right].$$

Assuming that the velocity field $\mathbf{u}^n$ is divergence-free, then it descends that the velocity field $\mathbf{u}^{n+1}$ has null divergence when the divergence of the right-hand side term is null. From these considerations, we derive a new Poisson equation for pressure:

$$\sum_{i=1}^{3} \frac{\delta}{\delta x_i} \left(\frac{\delta p^{n+1}}{\delta x_i} \right) = \sum_{i=1}^{3} \frac{\delta H_i^{n+1}}{\delta x_i}.$$

We observe that this equation cannot be solved for the pressure, because the velocity field required to compute the H-operator on the right hand side is still unknown. So

it is necessary to define a strategy, and a sort of predictor-corrector procedure is introduced. One idea might be, for example, to predict the velocity field from the momentum equation (4.10) neglecting the pressure term (this velocity field usually does not respect the divergence constraint), then the new pressure field is computed such that it corrects the predicted velocity and makes it divergence-free. The algorithmic steps related to this scheme are:

- Start at time t_n with the velocity field $\mathbf{u}^n$ assumed to respect the null-divergence constrain.

- Let $\mathbf{u}^*$ be the predicted velocity field which would result by neglecting the pressure contribute in the momentum equation:

$$\frac{\rho u_i^* - \rho u_i^n}{\Delta t} = H_i^* \quad \Longrightarrow \quad u_i^* = u_i^n + \frac{\Delta t}{\rho} H_i^*, \quad i = 1, \dots, 3. \tag{4.11}$$

  Notice that here the H-operator (defined in Eq. (4.3)) H_i^* is quadratic in the unknown $\mathbf{u}^*$ because it contains the advective flux term. A linearization similar to the one adopted in **Case C**, Eq. (4.9), is applied

$$\sum_{j=1}^{3} \frac{\delta(\rho u_j^* u_i^*)}{\delta x_j} \quad \overset{\text{(lineariz.)}}{\longrightarrow} \quad \sum_{j=1}^{3} \frac{\delta\left((\rho u_j^n) u_i^*\right)}{\delta x_j}, \quad i = 1, \dots, 3. \tag{4.12}$$

- Assume that the correct velocity field $\mathbf{u}^{n+1}$ respects the following equation

$$\begin{aligned} & \frac{\rho u_i^{n+1} - \rho u_i^n}{\Delta t} = H_i^* - \frac{\delta p^{n+1}}{\delta x_i}, \ i = 1, \dots, 3 \\ \Longleftrightarrow \ & u_i^{n+1} = u_i^n + \frac{\Delta t}{\rho}\left(H_i^* - \frac{\delta p^{n+1}}{\delta x_i}\right), \ i = 1, \dots, 3 \\ \overset{(4.11)}{\Longleftrightarrow} \ & u_i^{n+1} = u_i^* - \frac{\Delta t}{\rho}\frac{\delta p^{n+1}}{\delta x_i}, \ i = 1, \dots, 3. \end{aligned} \tag{4.13}$$

  One notes that requiring the velocity field $\mathbf{u}^{n+1}$ to be divergence-free means computing the new pressure field that satisfies the Poisson equation

$$\frac{\Delta t}{\rho}\sum_{i=1}^{3}\frac{\delta}{\delta x_i}\frac{\delta p^{n+1}}{\delta x_i} = \sum_{i=1}^{3}\frac{\delta u_i^*}{\delta x_i} \quad \overset{(4.11)}{\Longleftrightarrow} \quad \frac{\Delta t}{\rho}\sum_{i=1}^{3}\frac{\delta}{\delta x_i}\frac{\delta p^{n+1}}{\delta x_i} = \sum_{i=1}^{3}\frac{\delta}{\delta x_i}\left(u_i^n + \frac{\Delta t}{\rho}H_i^*\right). \tag{4.14}$$

  Since the velocity field $\mathbf{u}^n$ was assumed divergence free, and because of the linearity of the divergence operator, the previous equation is equivalent to the next one:

$$\sum_{i=1}^{3}\frac{\delta}{\delta x_i}\frac{\delta p^{n+1}}{\delta x_i} = \sum_{i=1}^{3}\frac{\delta H_i^*}{\delta x_i}. \tag{4.15}$$

  For consistency of the numerical scheme, the outer differential operators on both sides (that correspond to the numerical divergence applied to the momentum equation) must be discretized in the same way, that should be the same used for the numerical discretization on the continuity equation (4.2a).

- By using the new pressure field p^{n+1} computed from the Poisson equation (4.14) or (4.15), the divergence-free velocity field $\mathbf{u}^{n+1}$ is obtained in a straightforward way from Eq. (4.13).

- Pass to the next time step t_{n+1}.

Observe that the final fields $\mathbf{u}^{n+1}$ and p^{n+1} may not respect the original fully-implicit momentum equation (4.10) with the consequence that the pressure and velocity are not coupled properly and that the linearization is not properly resolved. More accurate schemes, like those described in the next section, overcome to this problem by using iterations. Algorithms like this, where the first velocity field computed (namely $\mathbf{u}^*$) does not satisfy the continuity equation and hence it is corrected by subtracting something, like the pressure gradient, are called ***projection methods***. The idea behind this name is that the divergence-producing part of the field is projected out.

We make some further observations about the cases just exposed and we highlight good and bad points of them. The schemes that adopt an explicit discretization of the pressure term, cases **A** and **C**, do not guarantee the new velocity to be divergence-free, neither they have an equation to move the pressure field forward in time. The *implicit treatment of the pressure field*, as done in cases **B** and **D**, requires an additional equation for it, which is a Poisson equation, derived by taking the divergence of the momentum equation. According to this, the resulting new velocity field respects the kinematic constrain. The advantages of having a divergence-free velocity and an equation to compute pressure make the schemes **B** and **D** better than **A** and **C**. From a further comparison, case **B** has the pro that the original full-discretized momentum equation is solved perfectly, but, being the H-operator treated in an explicit way, there is a strict condition on the time step. The fully implicit treatment is preferred because, being unconditionally stable, it allows using larger time steps, hence the scheme **D** is the best. Moreover, the algorithmic difficulty that rises from the fully implicit treatment is fixed by applying a predictor-corrector strategy to the *velocity* computation. Finally, the fact that the velocity and pressure fields computed may not verify the original momentum equation, which results in a loss of coupling, is overcome by introducing iterations, as described in the next section.

4.1.1 Implicit method with pressure correction

We pass from the intuitive notation adopted in the previous section to the notation introduced in Section §1.3, widespread in the OpenFOAM context. We recall that the spatial domain is discretized in $\mathcal{N}$ cells, in each cell a governing equation is numerically solved (by applying the discretization procedures explained in §1.3.2) and the equations form a system $\mathbf{Ax} = \mathbf{b}$ (as exposed in §1.3.3). Each equation may be written as in Eq. (1.143):

$$A_P x_P + \sum_N A_N x_N = b_P,$$

where the subscripts highlight whether the variables are evaluated in the centroid of the considered cell P, or in the centroids of the neighboring cells N. Since we solve the momentum equation, the unknowns x_P are $u_{i,P}$, namely the i-th component of the velocity vector referred to the centroid of the cell P. Moreover, we take apart the pressure term. In the framework of this notation, the discretized momentum equation referred to

the P-th control volume and to the i-th velocity component writes as follows:

$$A_P u_{i,P} + \sum_N A_N u_{i,N} = b_P - \left(\frac{\delta p}{\delta x_i}\right)_P, \; i = 1, \ldots, 3.$$

The pressure term retains the symbolic writing $\delta/\delta x_i$ because it underlines that the solution method exposed here is independent of the specific discretization scheme chosen for such term.

As observed previously, a direct solution of the equation is impossible, so the use of a predictor-corrector strategy, the derivation of a Poisson equation and the linearization of the quadratic term are applied. In the next sections we present three algorithms to solve the discretized momentum equation which mainly differ for the problems that they solve. In §4.1.1.1, we introduce the SIMPLE algorithm, used to solve steady-states problems, which adopts an iterative procedure to improve the errors rising from the linearization of the advective term. In §4.1.1.2 we present the PISO algorithm, that applies to the transient problems, which focuses on the improvement of the coupling between pressure and velocity. Even though PISO is usually referred to as an iterative method, the number of necessary steps are few and most of the times 2 or 3 steps are enough to have a good coupling. Both SIMPLE and PISO adopt the same **predictor-corrector** type of procedure for velocity and pressure, the only difference is the context in which they are applied: in the case of SIMPLE, the predictor-corrector computation happens in each iterative step, whereas, in the case of PISO, it is applied for each time step. The detailed derivation of the predictor-corrector scheme is exposed only once, within the description of the SIMPLE algorithm. Finally, in §4.1.1.3 we describe an hybrid scheme, implemented in OpenFOAM, that adopts the two strategies of SIMPLE and PISO, the so called PIMPLE algorithm, which manages to solve both the problems of coupling and linearization. PIMPLE applies to transient problems and considers each time step as a steady-state problem to solve, whereby it applies the SIMPLE scheme to improve the errors due to the linearization, and at each iteration the PISO scheme is employed to improve the coupling between velocity and pressure.

4.1.1.1 SIMPLE

The SIMPLE algorithm was developed in the early 1970s by Prof. Brian Spalding and his students [33]. The name stands for Semi-Implicit Method for Pressure Linked Equations, in fact the discretized momentum equation and the pressure correction equations are solved implicitly, whereas the velocity correction is treated explicitly. The SIMPLE algorithm was born to solve a steady-state problem for incompressible flows. The major aim of the algorithm is to resolve the non-linearity present in the advective term of the momentum equation, which is linearized in an iterative fashion. Being a method for steady-state problems, there is no time dependence and the iterations are necessary to reach the convergence to the solution for which the momentum equation is verified and agrees with the boundary conditions. At each iteration, the quadratic term is linearized with the velocity of the previous iteration step and a new divergence-free velocity and a new pressure are determined in such a way that these verify the linearized momentum equation.

At the m-th iterative step, we assume to know the velocity and pressure fields computed at the previous iterative step, $\mathbf{u}^{(m-1)}$ and $p^{(m-1)}$, and moreover velocity respects the continuity equation so it is divergence-free. The velocity field at the m-th iterative

step is **predicted** by using the last pressure field $p^{(m-1)}$ in the discretized momentum equation:

$$A_P u^*_{i,P} + \sum_N A_N u^*_{i,N} = b_P - \left(\frac{\delta p^{(m-1)}}{\delta x_i}\right)_P . \tag{4.16}$$

The advective flux term is linearized as follows

$$\sum_{j=1}^{3} \frac{\delta(\rho u^*_i u^*_j)}{\delta x_j} \quad \xrightarrow{\text{(lineariz.)}} \quad \sum_{j=1}^{3} \frac{\delta\left(\left(\rho u^{(m-1)}_j\right) u^*_i\right)}{\delta x_j}, \quad i = 1, \ldots, 3,$$

so that the OpenFOAM discretization of the advective term (see Eqs. (1.130) and (1.131)) is:

$$\sum_{f \in \partial V_P} \phi^{(m-1)}_f u^*_{i,f}, \quad \text{with} \quad \phi^{(m-1)}_f := \rho u^{(m-1)}_{i,f} \cdot \mathbf{S}_{f,P},$$

where the quantity ϕ_f denotes the volumetric flux of the advected momentum. The coefficients $\phi^{(m-1)}_f$ that result from the discretization of the advective term are contained in the matrices A_P and A_N (as shown in Section §1.3.3, in particular in Eq. (1.144)). When we want to underline the evaluation step of the volumetric flux ϕ_f we abound with the notation and write, for example, $A^{(m-1)}_P$ and $A^{(m-1)}_N$. The predicted velocity $\mathbf{u}^*$ solves the Eq. (4.16). We rearrange this equation as follows:

$$\begin{aligned} u^*_{i,P} &= \frac{1}{A_P}\left(-\sum_N A_N u^*_{i,N} + b_P\right) - \frac{1}{A_P}\left(\frac{\delta p^{(m-1)}}{\delta x_i}\right)_P \\ &=: \widetilde{u}^*_{i,P} - \frac{1}{A_P}\left(\frac{\delta p^{(m-1)}}{\delta x_i}\right)_P . \end{aligned} \tag{4.17}$$

We look for two correction fields $\mathbf{u}'$ and p', for the velocity and pressure respectively, in order to obtain the **corrected** velocity and pressure fields at the m-th iterative step:

$$u^{(m)}_{i,P} = u^*_{i,P} + u'_{i,P}, \tag{4.18}$$

$$p^{(m)}_P = p^{(m-1)}_P + p'_P \tag{4.19}$$

(where the previous pressure field is considered as the predicted field) such that the linearized momentum equation is respected:

$$A^{(m-1)}_P u^{(m)}_{i,P} + \sum_N A^{(m-1)}_N u^{(m)}_{i,N} = b_P - \left(\frac{\delta p^{(m)}}{\delta x_i}\right)_P . \tag{4.20}$$

Because of the linearity of the differential operators, of the discretization schemes, and of the relations (4.18) and (4.19), it descends the next momentum equation

$$A_P u^*_{i,P} + A_P u'_{i,P} + \sum_N A_N u^*_{i,N} + \sum_N A_N u'_{i,N} = b_P - \left(\frac{\delta p^{(m-1)}}{\delta x_i}\right)_P - \left(\frac{\delta p'}{\delta x_i}\right)_P .$$

Since $\mathbf{u}^*$ verifies Eq. (4.16), the previous equation is equivalent to the next relation between the corrections $\mathbf{u}'$ and p'

$$A_P u'_{i,P} + \sum_N A_N u'_{i,N} = -\left(\frac{\delta p'}{\delta x_i}\right)_P . \tag{4.21}$$

By rearranging the terms, we find the following relation:

$$\begin{aligned} u'_{i,P} &= -\frac{1}{A_P}\sum_N A_N u'_{i,N} - \frac{1}{A_P}\left(\frac{\delta p'}{\delta x_i}\right)_P \\ &=: \widetilde{u}'_{i,P} - \frac{1}{A_P}\left(\frac{\delta p'}{\delta x_i}\right)_P . \end{aligned} \tag{4.22}$$

Moreover we require that the corrected velocity is divergence free:

$$\sum_{i=1}^{3}\frac{\delta u^{(m)}_{i,P}}{\delta x_i} \overset{(4.18)}{=} \sum_{i=1}^{3}\frac{\delta u^{*}_{i,P}}{\delta x_i} + \sum_{i=1}^{3}\frac{\delta u'_{i,P}}{\delta x_i} = 0,$$

and by using the relations in Eqs. (4.17) and (4.22) relative to the expression of $u^*_{i,P}$ and $u'_{i,P}$, we find

$$\sum_{i=1}^{3}\frac{\delta \widetilde{u}^{*}_{i,P}}{\delta x_i} - \sum_{i=1}^{3}\frac{\delta}{\delta x_i}\left[\frac{1}{A_P}\left(\frac{\delta p^{(m-1)}}{\delta x_i}\right)\right]_P + \sum_{i=1}^{3}\frac{\delta \widetilde{u}'_{i,P}}{\delta x_i} - \sum_{i=1}^{3}\frac{\delta}{\delta x_i}\left[\frac{1}{A_P}\left(\frac{\delta p'}{\delta x_i}\right)\right]_P = 0.$$

Because of the linearity of the discretized gradient operator, we may use the definition of $p^{(m)}$ of Eq. (4.19) into the previous equation and find the next relation between the known field $\widetilde{u}^*_i$ and the unknowns $\widetilde{u}'_i$ and $p^{(m)}$:

$$\sum_{i=1}^{3}\frac{\delta \widetilde{u}^{*}_{i,P}}{\delta x_i} - \sum_{i=1}^{3}\frac{\delta}{\delta x_i}\left[\frac{1}{A_P}\left(\frac{\delta p^{(m)}}{\delta x_i}\right)\right]_P + \sum_{i=1}^{3}\frac{\delta \widetilde{u}'_{i,P}}{\delta x_i} = 0. \tag{4.23}$$

At this point, the classic version of the SIMPLE algorithm neglects the contribution of the field $\widetilde{u}'_i$ and solve the following Poisson equation to determine the pressure field $p^{(m)}$:

$$\sum_{i=1}^{3}\frac{\delta}{\delta x_i}\left[\frac{1}{A_P}\left(\frac{\delta p^{(m)}}{\delta x_i}\right)\right]_P = \sum_{i=1}^{3}\frac{\delta \widetilde{u}^{*}_{i,P}}{\delta x_i}. \tag{4.24}$$

By looking at the Eq. (4.17) of $u^*_{i,P}$, we can interpret the right hand side term of Eq. (4.24) as the continuity error of the velocity field $\mathbf{u}^*$ without the pressure gradient; it is given by the fluxes summed and interpolated through the faces. By solving this Poisson equation, the new pressure field $p^{(m)}$ is determined. The new velocity field respects the following relation

$$\begin{aligned} u^{(m)}_{i,P} &\overset{(4.18)}{=} u^{*}_{i,P} + u'_{i,P} \\ &\overset{(4.17)}{=} \widetilde{u}^{*}_{i,P} - \frac{1}{A_P}\left(\frac{\delta p^{(m-1)}}{\delta x_i}\right)_P + u'_{i,P} \\ &\overset{(4.22)}{=} \widetilde{u}^{*}_{i,P} - \frac{1}{A_P}\left(\frac{\delta p^{(m-1)}}{\delta x_i}\right)_P + \widetilde{u}'_{i,P} - \frac{1}{A_P}\left(\frac{\delta p'}{\delta x_i}\right)_P , \end{aligned}$$

and by using the linearity of the discretized gradient operator and neglecting the contribution of the field $\widetilde{u}'_i$ as previously done, we find the final relation to compute the corrected velocity field:

$$u^{(m)}_{i,P} = \widetilde{u}^{*}_{i,P} - \frac{1}{A_P}\left(\frac{\delta p^{(m)}}{\delta x_i}\right)_P . \tag{4.25}$$

The corrected fields respect the momentum Eq. (4.20) where the flux is linearized with the velocity at the previous iteration. Then the fluxes are updated by using the new velocity field (so the matrices A_P and A_N are updated too) and we must check the updated momentum equation

$$A_P^{(m)} u_{i,P}^{(m)} + \sum_N A_N^{(m)} u_{i,N}^{(m)} = b_P - \left(\frac{\delta p^{(m)}}{\delta x_i} \right)_P , \tag{4.26}$$

that is only approximated due to the terms neglected so far. If the momentum Eq. (4.26) is respected within a certain prescribed tolerance, then the algorithm has reached the convergence and the problem due to the linearization is solved, otherwise the algorithm must continue with at least another iteration.

To sum up, at each m-th iteration of the SIMPLE algorithm the following steps are done:

1. the predicted velocity field $\mathbf{u}^*$ is computed by Eq. (4.16),
2. the field $\widetilde{\mathbf{u}}^*$ defined in Eq. (4.17) is computed,
3. the new corrected pressure field $p^{(m)}$ is computed by solving the Poisson Eq. (4.24),
4. the new corrected velocity field $\mathbf{u}^{(m)}$ is computed by Eq. (4.25),
5. the momentum matrix (namely $\mathbf{A}$) is assembled and the advective terms are linearized by using the last velocity field determined $\mathbf{u}^{(m)}$, so that $A_P = A_P^{(m)}$ and $A_N = A_N^{(m)}$,
6. check if the updated momentum Eq. (4.26) is verified within a certain tolerance and check if the number of iterations done is less than the maximum number of iterations fixed a priori; if it is not, continue with a new iterative step.

From the numerical point of view, remember to pay the usual attention to the discretization of the pressure equation (4.24): the inner derivative of pressure must be discretized in the same way it is treated in the momentum equation (4.16); the outer derivatives (on both left and right hand sides of the equations) must be discretized by the same scheme.

Further considerations. Even though the scheme derived makes use of the corrector fields $\mathbf{u}'$ and p', from the practical point of view, they are never computed. Neglecting the velocity correction $\widetilde{\mathbf{u}}'$ in the SIMPLE algorithm (in the Poisson Eq. (4.23) for the computation of pressure), despite it is a common practice, is almost a crudeness and probably this is also the major reason why the method is slow to converge. Some variations that treat more kindly $\widetilde{u}'$ or that improve the convergence are mentioned below, but without entering in the details.

- The SIMPLEC (i.e. SIMPLE Consistent) method, by van Doormal and Raithby (1984) [260], see the approximation of $u'_{i,P}$ as a weighted mean of the neighbor values; this permits to approximate $\widetilde{u}'_{i,P}$ in terms of u'. In this way the Eq. (4.22) changes its expression only in terms of u' and p'.

- The SIMPLER (i.e. SIMPLE Revised) method was proposed by Patankar in 1980 [207]. For it, the term $\widetilde{u}'_{i,P}$ is neglected, as in SIMPLE, but the pressure correction p' derived is used only to determine the correction u' and to correct the velocity field $u^{(m)}$, it is not used to correct pressure $p^{(m-1)}$. In fact, having the new velocity field $u^{(m)}$, the pressure field $p^{(m)}$ may be determined by Eq. (4.24) where $\widetilde{u}^*$ is replaced by $\widetilde{u}^{(m)}$.

In OpenFOAM, the SIMPLE algorithm (together with the possibility to apply the under-relaxation) and the SIMPLEC algorithm are implemented for solvers applicable only to steady-states problems. Missing the transient term, the time step does not have the usual meaning, then the under-relaxations are important to gain in stability and convergence. Setting the time step equal to 1, the simulation time corresponds to the number of iterations of the SIMPLE loop. Setting the relaxation coefficient very small surely improves a lot the stability, but, on the other hand, it may greatly extend the time required for convergence. Moreover, the solution may respect the convergence condition even if the steady-state has not been reached.

4.1.1.2 PISO

The Pressure Implicit with the Splitting of Operators (PISO) is an algorithm introduced by Issa [134]. The PISO algorithm is implemented in a solver devoted to transient problems and uses a predictor-corrector strategy similar to the SIMPLE scheme, but, unlike that, PISO puts some effort into improving the velocity and pressure coupling by using multiple corrector steps. Even though PISO is usually considered as an iterative scheme, just two or three corrective steps are recommended most of the time. We show the procedure for the case of two corrective steps without entering in the details of the predictor-corrector strategy because it was explained in the previous section §4.1.1.1. We consider the $(n+1)$-st time step, then the fields $\mathbf{u}^n$ and p^n derived at the previous time step are known.

Predictor Step. The momentum matrix is assembled (and so even the vector of the known terms b_P) and the advective quadratic term is linearized, then the volumetric flux ϕ_f is calculated by using the known field $\mathbf{u}^n$ and we associate the superscript to the matrices $A_P = A_P^n$ and $A_N = A_N^n$. We notice that the fluxes ϕ_f are determined by a velocity field $\mathbf{u}^n$ which is divergence-free (namely it respects the continuity equation). The predicted velocity field $\mathbf{u}^*$ (namely the prediction of $\mathbf{u}^{n+1}$) is computed by solving the following momentum equation

$$A_P u^*_{i,P} + \sum_N A_N u^*_{i,N} = b_P - \left(\frac{\delta p^n}{\delta x_i}\right)_P .$$

The field $\widetilde{\mathbf{u}}^*$ is determined again as

$$\widetilde{u}^*_{i,P} := \frac{1}{A_P}\left(-\sum_N A_N u^*_{i,N} + b_P\right),$$

and it is used to compute the first corrected pressure field.

1^{st} Corrector step. We search two correction fields $\mathbf{u}'$ and p' for velocity and pressure in order to obtain the corrected fields

$$u_{i,P}^{(1)} = u_{i,P}^{*} + u_{i,P}',$$
$$p_P^{(1)} = p_P^n + p_P'.$$

As we have seen in the previous section, the two correction fields $\mathbf{u}'$ and p' are not determined explicitly, but these relations are useful to determine the corrected fields. The first corrected pressure $p^{(1)}$ is obtained by solving the following Poisson equation

$$\sum_{i=1}^{3} \frac{\delta}{\delta x_i} \left[\frac{1}{A_P} \left(\frac{\delta p^{(1)}}{\delta x_i} \right) \right]_P = \sum_{i=1}^{3} \frac{\delta \widetilde{u}_{i,P}^{*}}{\delta x_i}.$$

With this corrected pressure, we obtain the first corrected velocity field $\mathbf{u}^{(1)}$ as

$$u_{i,P}^{(1)} = \widetilde{u}_{i,P}^{*} - \frac{1}{A_P} \left(\frac{\delta p^{(1)}}{\delta x_i} \right)_P.$$

The corrected velocity field $\mathbf{u}^{(1)}$ is divergence free by construction. We also notice that the corrected pressure field was determined by using a pressure field that is not conservative (in fact, $\mathbf{u}^*$ may not respect the continuity equation with the divergence-free condition). Moreover, from the previous equation it descends that the new corrected fields verify the following momentum equation:

$$A_P u_{i,P}^{(1)} + \sum_N A_N u_{i,N}^{*} = b_P - \left(\frac{\delta p^{(1)}}{\delta x_i} \right)_P.$$

2^{nd} Corrector step. We again look for two correction fields $\mathbf{u}''$ and p'' for velocity and pressure (which are not directly computed) in order to obtain the new corrected fields

$$u_{i,P}^{(2)} = u_{i,P}^{(1)} + u_{i,P}'',$$
$$p_P^{(2)} = p_P^{(1)} + p_P''.$$

We pass to the second step of correction, therefore we need to calculate $\widetilde{\mathbf{u}}^{(1)}$:

$$\widetilde{u}_{i,P}^{(1)} := \frac{1}{A_P} \left(-\sum_N A_N u_{i,N}^{(1)} + b_P \right),$$

which is used to compute the new corrected pressure by solving the Poisson equation

$$\sum_{i=1}^{3} \frac{\delta}{\delta x_i} \left[\frac{1}{A_P} \left(\frac{\delta p^{(2)}}{\delta x_i} \right) \right]_P = \sum_{i=1}^{3} \frac{\delta \widetilde{u}_{i,P}^{(1)}}{\delta x_i}.$$

Having the second corrected pressure, we compute the second corrected velocity $\mathbf{u}^{(2)}$

$$u_{i,P}^{(2)} = \widetilde{u}_{i,P}^{(1)} - \frac{1}{A_P} \left(\frac{\delta p^{(2)}}{\delta x_i} \right)_P.$$

Therefore the new corrected fields verify the following equation:

$$A_P u_{i,P}^{(2)} + \sum_N A_N u_{i,N}^{(1)} = b_P - \left(\frac{\delta p^{(2)}}{\delta x_i}\right)_P .$$

We underline that, in this second corrector step, the corrected pressure $p^{(2)}$ was computed on the basis of a conservative velocity field (because the velocity field $\mathbf{u}^{(1)}$, used to form $\widetilde{\mathbf{u}}^{(1)}$, is divergence-free, hence it respects the continuity equation).

The PISO correction procedure may stop in two cases: if the number of iterations reaches the maximum number of iterations fixed a priori, or if the continuity equation is verified (within a certain tolerance) by the corrected pressure and velocity fields. In order to understand the last statement, consider the last corrected fields that we determined $\mathbf{u}^{(2)}$ and $p^{(2)}$; we could say that these fields satisfy the continuity equation if they verify (within a certain tolerance) the following Poisson equation

$$\sum_{i=1}^{3} \frac{\delta}{\delta x_i}\left[\frac{1}{A_P}\left(\frac{\delta p^{(2)}}{\delta x_i}\right)\right]_P = \sum_{i=1}^{3} \frac{\delta \widetilde{u}_{i,P}^{(2)}}{\delta x_i}, \tag{4.27}$$

where

$$\widetilde{u}_{i,P}^{(2)} := \frac{1}{A_P}\left(-\sum_N A_N u_{i,N}^{(2)} + b_P\right).$$

Owning the transient term, the condition on the time step is necessary to ensure the stability and the CFL condition $c < 1$ must be respected (see §1.2.2.2).

4.1.1.3 PIMPLE

OpenFOAM offers another algorithm for transient problems that merges the two schemes, SIMPLE and PISO, and for this reason it is called PIMPLE. At each time step, PIMPLE uses the SIMPLE iteration, looking for a steady-state solution, and it moves on the next time step when the convergence is achieved. Moreover, at every iteration of SIMPLE, inner iterations for the pressure correction are applied by the adoption of the PISO algorithm. Because of such nested iterations structure, the SIMPLE iterations are referred to as the **outer loop**, whereas the PISO iterations are called **inner loop**. Figure 4.1 shows a synthetic diagram of the loops.

Listing 4.1 reports an example of user defined instructions for the PIMPLE algorithm. The `nOuterCorrectors` parameter defines the maximum number of outer loops and `nCorrectors` the maximum number of the inner loops. The number of outer iterations recommended by the Guide is between 50 and 1000, whereas the default is 1 (namely a setting for which PIMPLE runs in PISO mode). In the case the mesh is non orthogonal, the user can set the number of times the pressure equation is computed and corrected in order to reduce the influence of a bad computational mesh by using `nNonOrthogonalCorrectors`. In addition to the maximum number of iterations, there is a stop criterion based on the residuals. It is possible setting a specific value of tolerance for each equation. The absolute tolerance is by default 10^{-5}, but the user can change it. In fact, the user can specify whether to use a condition on the relative residuals (by setting a non-zero value for the parameter `relTol`) or the absolute residuals (by setting a non-zero value for the parameter `tolerance`); an example is reported in Listing 4.1 lines 94–101, where a control on the absolute tolerance is set to 10^{-2} for both pressure and velocity equations.

```
PIMPLE
{
    momentumPredictor   yes;
    nOuterCorrectors    50;
    nCorrectors         2;
    nNonOrthogonalCorrectors 0;

    residualControl
    {
        "(p|U)"
        {
            tolerance 1e-2;
            relTol 0;
        }
    }
}
```

Listing 4.1: Example of setting parameters for the PIMPLE algorithm, written in the `fvSolution` file.

For steady-state problems, SIMPLE is preferred to PISO because, in such case, it is not necessary to solve the pressure-velocity coupling. For transient problems, both PISO and PIMPLE may be used, but PIMPLE ensures a better stability (when the outer loop is effectively computed). PIMPLE permits to have multiple outer and inner iterations and it takes the benefits of the other two schemes, allowing to use the CFL condition $c \geqslant 1$ (whereas PISO requires $c < 1$ for stability).We also highlight that the SIMPLE part of the PIMPLE algorithm may be computed in the SIMPLEC (SIMPLE consistent) mode.

The solver that we adopt in our tests of §4.7 is a modification of `interFoam`, which is a scheme already available in OpenFOAM to solve the incompressible multiphase problems and implements exactly the PIMPLE scheme. In §4.5 we give a wide overlook at our scheme called `interThermalRadConvFoam`, underlining the structure of the PIMPLE algorithm adopted. In the next sections, instead, we focus on the schemes to solve the equation for the transport of the phase volumetric fraction which is characteristic of the VoF method.

4.2 Multiphase flows: VOF method and Flux-Corrected Transport scheme

The multiphase dynamics of our model is treated by the Volume of Fluid method that considers the variable describing the phase volumetric fraction α and derives a transport equation for that. The liquid phase corresponds to $\alpha = 1$, whereas the gaseous phase to $\alpha = 0$, and the phase interface between liquid and gas is determined by the gradient of α, therefore the numerical schemes that discretize the α equation must keep the values bounded between 0 and 1 and the phase interface sharp. Our model numerically solves a modified version of this transport equation by using the Flux-Corrected Technique that was introduced by Boris and Book [22] in order to manage the steep gradients well. The idea behind this technique is to take advantage of the good properties of both low and high order schemes (that we discussed in §1.2), alleviating the problems that both of them entail. In fact, the low order schemes, like the upwind scheme, guarantee the boundedness of the solutions, but, on the other hand, lead to the smearing of the phase interface because they are diffusive. Vice versa, by employing high order schemes there are

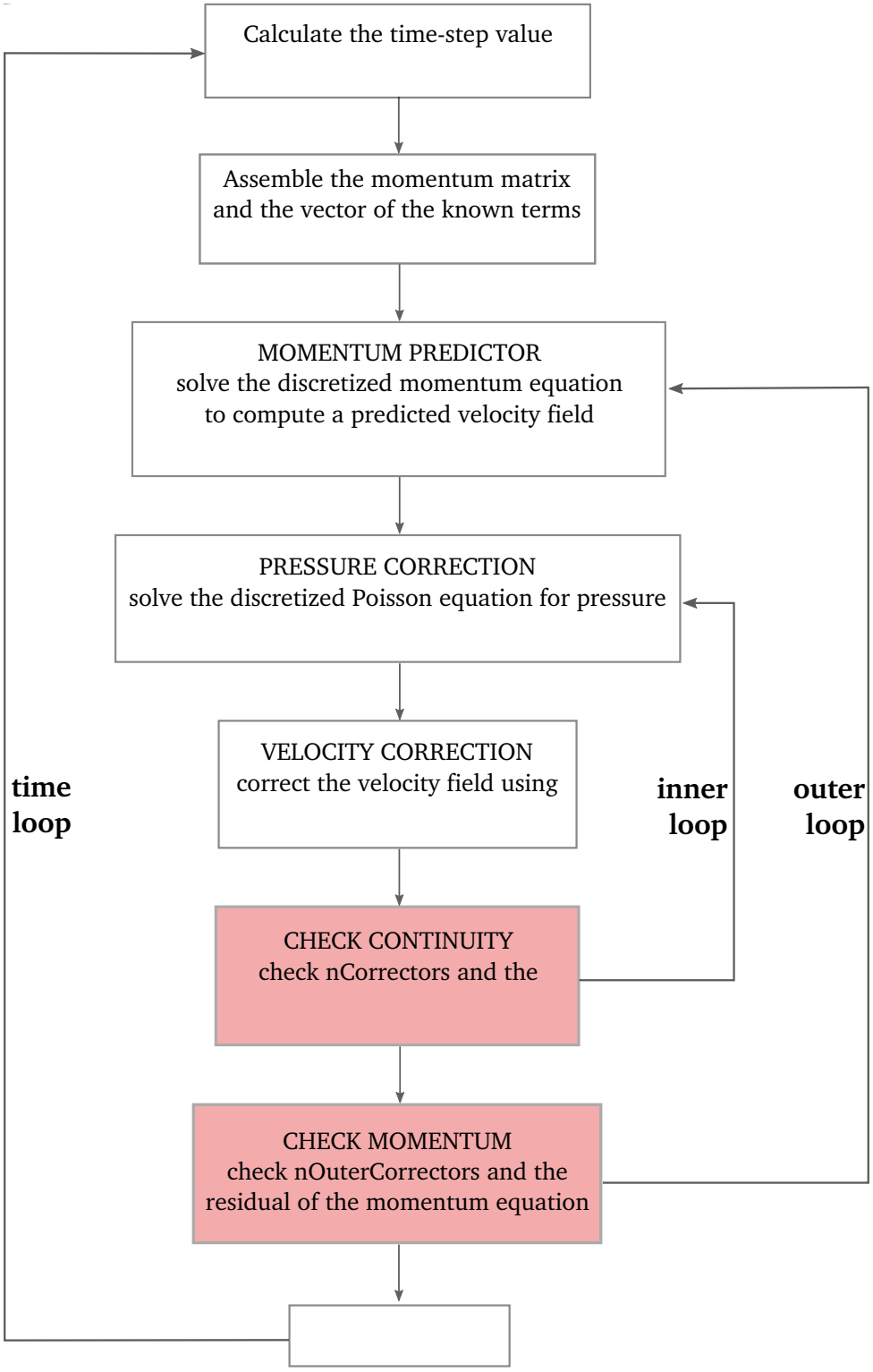


Figure 4.1: Scheme of the *inner* and *outer* loops in PIMPLE.

no longer problems of diffusion, but they carry the risk of introducing new maximum and minimum; for this reason, flux limiters are adopted passing to the use of TVD schemes. Besides, high order schemes require a greater computational cost with respect to the low order schemes. The flux-corrected technique proposes a mix of both kinds of schemes, breaking down their problems and combining their benefits. We pass to describe how the FCT technique is applied in the discretization of the α equation in the framework of

OpenFOAM.

The semi-discretization of α-equation is considered, where the spatial discretization of the advective term is obtained by using the Gauss Theorem and an interpolation scheme (see the details of the space discretization in the OpenFOAM framework in §1.3.2.1), hence we have to solve, for any cell V_P of the discretized domain, the following equation for α_P (where α_P is the volume integral of α over V_P):

$$\frac{\partial \alpha_P}{\partial t} = -\frac{1}{|V_P|} \sum_{f \in \partial V_P} F_f,$$

where F_f denotes the total flux related to the f-face. The key steps required for the computation of the numerical flux adopted by FCT technique are:

1. Compute the numerical flux $F_{f,U}$ by a low order scheme,
2. Compute the numerical flux $F_{f,H}$ by a high order scheme,
3. Define the anti-diffusive flux
$$A_f := F_{f,H} - F_{f,U}, \tag{4.28}$$
4. Compute the *corrected* flux:
$$\begin{aligned} F_{f,C} &= \lambda_M F_{f,H} + (1 - \lambda_M) F_{f,U} \\ &= F_{f,U} + \lambda_M A_f, \quad \lambda_M \in [0, 1], \end{aligned} \tag{4.29}$$
The corrected flux is a low order flux modified by a high order anti-diffusive flux limited by the weighting factor λ_M (we had introduced this procedure of flux limiters in §1.2.7). The weighting factor λ_M is active only at the interface, taking the value $\lambda_M = 1$, so that the high order scheme works at the interface to preserve the sharpness, whereas $\lambda_M = 0$ otherwise and the advection is treated in a straightforward way with the upwind scheme; this trick not only reduces the numerical diffusion at the interface but also permits to lower the computational cost of the scheme [69].
5. Solve the equation:
$$\frac{\partial \alpha_P}{\partial t} = -\frac{1}{|V_P|} \sum_{f \in \partial V_P} F_{f,C}. \tag{4.30}$$

The FCT scheme is determined when the interpolation schemes for steps 1 and 2 above are fixed and once decided how to compute the limiter λ_M. In OpenFOAM, the corrected flux and the weighting factor λ_M are computed iteratively using the MULES algorithm (as we will show in section §4.3).

4.2.1 Low order, anti-diffusive, and compressive fluxes

We explore closely the different fluxes introduced before. First of all, we recall that the total flux at the face F_f is, according to Eq. (1.131), the product of the volumetric flux through the face ϕ_f and the value of the advected variable interpolated on the face α_f, i.e.:

$$F_f = \phi_f \alpha_f, \qquad \text{with} \qquad \phi_f := \mathbf{u}_f \cdot \mathbf{S}_f,$$

where $\mathbf{S}_f$ is the surface area vector, and $\mathbf{u}_f$ and α_f are obtained with interpolation schemes for the velocity and for α.

The low order flux F_U adopts the upwind scheme to determine the values α_f, so we write it as follows

$$F_{f,U} = \phi_f \alpha_{f,upwind}. \tag{4.31}$$

Instead, the high-order flux F_H uses a scheme that can be set by the user and hence we refer to it as $\alpha_{f,div}$. Since this term is active at the interface and the interface must be as sharp as possible, a fictitious term forcing the compression of the interface is added. The complete expression of $F_{f,H}$ is then

$$F_{f,H} = \phi_f \alpha_{f,div} + \phi_{rf} \alpha_{rf}(1 - \alpha_{rf}), \tag{4.32}$$

where ϕ_{rf} refers to the so-called compressive flux and α_{rf} is the value of the face interpolation of α. The term $\phi_{rf}\alpha_{rf}(1-\alpha_{rf})$ in Eq. (4.32) is an artificial flux added to keep the interface sharp since it pushes perpendicularly to the interface following the gradient of α. The compressive flux ϕ_{rf} is defined as follows:

$$\phi_{rf} = C_\alpha \frac{|\phi_f|}{|\mathbf{S}_f|} n_f. \tag{4.33}$$

The coefficient C_α in Eq. (4.33) is a parameter whose aim is to limit the interface smearing. C_α assumes values greater than or equal to zero and when it is set to zero means that the high order flux of Eq. (4.32) neglects the additional artificial term for the interface compression:

$$C_\alpha = 0 \implies F_{f,H} = \phi_f \alpha_{f,div}.$$

The term n_f in Eq. (4.33) is the unit flux normal to the face $\mathbf{S}_f$

$$n_f = \hat{\mathbf{n}}_f \cdot \mathbf{S}_f = \frac{(\nabla\alpha)_f}{|(\nabla\alpha)_f + \delta_N|} \cdot \mathbf{S}_f \tag{4.34}$$

where $\hat{\mathbf{n}}_f$ gives the compressive direction and δ_N is a desingularization factor to avoid the division by zero:

$$\delta_N = \frac{10^{-8}}{\left(\frac{1}{\mathcal{N}} \sum_i |V_i|\right)^{1/3}},$$

being $\mathcal{N}$ the number of computational cells V_i.

Finally, we write the complete expression of the corrected flux:

$$\begin{aligned} F_{f,C} &\overset{(4.29)}{=} F_{f,U} + \lambda_M A_f \\ &= F_{f,U} + \lambda_M (F_{f,H} - F_{f,U}) \\ &= \phi_f \alpha_{f,upwind} + \lambda_M \left[\phi_f \alpha_{f,div} + \phi_{rf}\alpha_{rf}(1-\alpha_{rf}) - \phi_f \alpha_{f,upwind}\right]. \end{aligned}$$

The user can select the interpolation schemes for $\alpha_{f,div}$ and α_{rf} by modifying the case file case/system/fvSchemes as shown in the following example (referring the reader to the Listing 1.1 for another example of file fvSchemes enriched by comments and to §A.1 for an overview on the organization and structure of OpenFOAM and its folders for the simulation tests)

```
divSchemes
{
    div(phi,alpha)   Gauss Gamma 1;
    div(phirb,alpha) Gauss linear;
    ...
}
```

where

$$\texttt{phi} \longleftrightarrow \phi_f,$$
$$\texttt{phirb} \longleftrightarrow \phi_{rf}.$$

Also, we observe that when the scheme chosen for $\alpha_{f,div}$ is exactly `upwind`, the anti-diffusive flux (defined in Eq. (4.28)) reduces only to the compressive flux, in fact:

$$\begin{aligned} A_f &= F_{f,H} - F_{f,U} \\ &= \phi_f \alpha_{f,upwind} + \phi_{rf}\alpha_{rf}(1-\alpha_{rf}) - \phi_f \alpha_{f,upwind} \\ &= \phi_{rf}\alpha_{rf}(1-\alpha_{rf}). \end{aligned}$$

In addition, the user can set the value for the compression coefficient C_α that is used in Eq. (4.33) by modifying the value attributed to `cAlpha` in the case file case/system/controlDict.

The unsteady term of Eq. (4.30) can be discretized only by implicit Euler or Crank-Nicolson differencing schemes, and solving the ODE equation $\partial_t \alpha_P = \mathcal{F}(\alpha_P)$ with those schemes leads to the next discretizations:

$$\text{Implicit Euler:} \quad \frac{\alpha_P^{n+1} - \alpha_P^n}{\Delta t} = \mathcal{F}(\alpha_P^{n+1}),$$
$$\text{Crank-Nicolson:} \quad \frac{\alpha_P^{n+1} - \alpha_P^n}{\Delta t} = \frac{1}{2}\mathcal{F}(\alpha_P^{n+1}) + \frac{1}{2}\mathcal{F}(\alpha_P^n).$$

When employed for complex flows in complex geometries, the Crank-Nicolson scheme can suffer of instability, hence an "off-centering" procedure is adopted to stabilize it. Thanks to this shrewdness, the Crank-Nicolson scheme maintains greater accuracy than the first-order Euler scheme. The solver `interFoam` offers the possibility to use a generalization of the Crank-Nicolson scheme (that is known as *Theta method* in the context of ODEs), which is implemented as a temporal blending with the coefficient θ_{CN}:

$$\frac{\alpha_P^{n+1} - \alpha_P^n}{\Delta t} = \theta_{CN}\mathcal{F}(\alpha_P^{n+1}) + (1-\theta_{CN})\mathcal{F}(\alpha_P^n). \tag{4.35}$$

The blending coefficient depends on the "off-center" coefficient, denoted as `ocCoeff`, that the user can set by modifying the number next to Crank-Nicolson specification in the temporal discretization schemes (that are written in the case file case/system/fvSchemes) as in the example reported below:

```
ddtSchemes
{
    ddt(alpha)      CrankNicolson 0.9;
    default         Euler;
}
```

where the Crank-Nicolson scheme, with `ocCoeff` = 0.9, is selected for α-equation, and the Euler scheme is the default choice for the remaining equations. The blending coefficient θ_{CN}, that is denoted as `cnCoeff`, is related to the off-center coefficient according to the following expression:

$$\texttt{cnCoeff} \longleftrightarrow \theta_{CN},$$
$$\texttt{cnCoeff} = \frac{1}{1 + \texttt{ocCoeff}}, \quad \texttt{ocCoeff} \in [0, 1].$$

If `ocCoeff` = 1, the blended scheme written in Eq. (4.35) is centered and consists of the classic Crank-Nicolson, instead, if `ocCoeff` = 0, the blended scheme coincides with implicit Euler. We highlight that, when the `Euler` scheme is used for the time discretization of the α equation, the off-centering coefficient is set equal to 0 by default to be used inside the solver file VoF/alphaEqn.H.

The latest thing that misses in order to have a completely determined numerical scheme for the α equation is the computation of the limiter λ_M through the use of the MULES scheme. The first version of the MULES scheme that has been presented in literature is an *explicit* algorithm that satisfies the following equation

$$\begin{aligned}
{\alpha_P}^{n+1} &= \alpha_P^n - \frac{\Delta t}{|V_P|} \sum_{f \in \partial V_P} F_{f,C}^n \\
&= \alpha_P^n - \frac{\Delta t}{|V_P|} \sum_{f \in \partial V_P} \left[F_{f,U}^n + \lambda_M A_f^n \right] \\
&= \alpha_P^n - \frac{\Delta t}{|V_P|} \sum_{f \in \partial V_P} \Big\{ \phi_f^n \alpha_{f,upwind}^n \\
&+ \lambda_M \Big[\phi_f^n \alpha_{f,div}^n + \phi_{rf}^n \alpha_{rf}^n (1-\alpha)_{rf}^n - \phi_f^n \alpha_{f,upwind}^n \Big] \Big\},
\end{aligned}$$

and where the limiter λ_M is computed by a inner iteration. Such a factor has to limit the whole anti-diffusive flux, details are shown in §4.4. This version is *explicit* and, for this reason, it imposes restrictions on the CFL conditions and, as a consequence, limits on the time step.

Instead, the latest variant of the algorithm proposed in literature follows a *semi-implicit predictor-corrector* approach where the operator splitting is coupled with the application of the MULES limiter to the explicit correction, rather than to the complete flux. An *implicit* predictor step is executed by considering only the low-order flux term, which ensures the boundedness, and then an *explicit correction* is built on which the MULES limiter is applied. According to this procedure, the predictor step is defined by the following expression:

$$\begin{aligned}
\text{Predictor: } \widetilde{\alpha}_P^{n+1} &= \alpha_P^n - \frac{\Delta t}{|V_P|} \sum_{f \in \partial V_P} \widetilde{F}_{f,U}^{n+1} \\
&= \alpha_P^n - \frac{\Delta t}{|V_P|} \sum_{f \in \partial V_P} \phi_f^n \widetilde{\alpha}_{f,upwind}^{n+1},
\end{aligned}$$

and we remark that the values at the cell faces of the volume fraction function α_f are actually expressed in terms of the cell-centered values α_P and α_N (as seen in §1.3.2.2), leading to a system of equations in the unknowns $\widetilde{\alpha}_P^{n+1}$ over all the control volumes. The

predicted values $\widetilde{\alpha}_P^{n+1}$ are used by the schemes `div(phi,alpha)` and `div(phirb,alpha)` to compute the values at the cell faces $\widetilde{\alpha}_{f,div}^{n+1}$ and $\widetilde{\alpha}_{rf}^{n+1}$ necessary for the explicit correction step:

$$
\begin{aligned}
\text{Corrector: } \alpha_P^{n+1} &= \alpha_P^n - \frac{\Delta t}{|V_P|}\sum_{f\in\partial V_P}\widetilde{F}_{f,C}^{n+1} \\
&= \alpha_P^n - \frac{\Delta t}{|V_P|}\sum_{f\in\partial V_P}\left[\widetilde{F}_{f,U}^{n+1} + \lambda_M \widetilde{A}_f^{n+1}\right] \\
&= \alpha_P^n - \frac{\Delta t}{|V_P|}\sum_{f\in\partial V_P}\Big\{\phi_f^n\widetilde{\alpha}_{f,upwind}^{n+1} \\
&+ \lambda_M\Big[\phi_f^n\widetilde{\alpha}_{f,div}^{n+1} + \phi_{rf}^n\widetilde{\alpha}_{rf}^{n+1}(1-\widetilde{\alpha}_{rf}^{n+1}) - \phi_f^n\widetilde{\alpha}_{f,upwind}^{n+1}\Big]\Big\} \\
&= \widetilde{\alpha}_P^{n+1} - \frac{\Delta t}{|V_P|}\sum_{f\in\partial V_P}\lambda_M\Big[\phi_f^n\widetilde{\alpha}_{f,div}^{n+1} + \phi_{rf}^n\widetilde{\alpha}_{rf}^{n+1}(1-\widetilde{\alpha}_{rf}^{n+1}) - \phi_f^n\widetilde{\alpha}_{f,upwind}^{n+1}\Big].
\end{aligned}
$$

According with this predictor-corrector scheme, the MULES limiter is computed on the anti-diffusive flux, but with an iterative procedure, details are shown in §4.3.

The user can decide to switch from the latest semi-implicit MULES method to the old explicit one by simply setting the variable `MULESCorr` as `true` instead of `false` inside the test case file case/system/fvSolutions. In the following two sections, we present in details both algorithms, in particular we go through the alphaEqn.H file located inside the multiphase/VoF solver folder. Moreover, we will see how the Crank-Nicolson time blending adoption is used in both cases and the different implementations of the diffusive flux inside the anti-diffusive flux A_f.

4.3 MULES limiter to explicit correction

We analyze the algorithm obtained by activating the `MULESCorr` procedure which adopts the predictor-corrector semi-implicit strategy by going through the alphaEqn.H file that contains the implementation (at a high level) of the scheme that solves α-equation.

4.3.1 Initial setting

Initially, some background variables are set. The names `alphaScheme` and `alpharScheme` are introduced to indicate the schemes that compute the values at the faces of the volume fraction variables in the Eq. (4.32)

```
2     word alphaScheme("div(phi,alpha)");
3     word alpharScheme("div(phirb,alpha)");
```

The off-centering coefficient `ocCoeff` is initialized, according to the temporal scheme chosen by the user: if Euler was selected, `ocCoeff`=0, otherwise the value of `ocCoeff` is specified by the user. After, the off-centering coefficient `ocCoeff` is used to compute the Crank-Nicolson blending coefficient `cnCoeff`.

```
5     // Set the off-centering coefficient according to ddt scheme
6     scalar ocCoeff = 0;
7     {
8         tmp<fv::ddtScheme<scalar>> tddtAlpha
9         (
```

```
            fv::ddtScheme<scalar>::New
            (
                mesh,
                mesh.ddtScheme("ddt(alpha)")
            )
        );
        const fv::ddtScheme<scalar>& ddtAlpha = tddtAlpha();

        if
        (
            isType<fv::EulerDdtScheme<scalar>>(ddtAlpha)
         || isType<fv::localEulerDdtScheme<scalar>>(ddtAlpha)
        )
        {
            ocCoeff = 0;
        }
        else if (isType<fv::CrankNicolsonDdtScheme<scalar>>(ddtAlpha))
        {
```

```
            {
                ocCoeff =
                    refCast<const fv::CrankNicolsonDdtScheme<scalar>>(
    ddtAlpha)
                   .ocCoeff();
            }
        }
        else
        {
            FatalErrorInFunction
                << "Only Euler and CrankNicolson ddt schemes are
    supported"
                << exit(FatalError);
        }
    }

    // Set the time blending factor, 1 for Euler
    scalar cnCoeff = 1.0/(1.0 + ocCoeff);
```

A new field called `phic` is introduced which refers the first argument of the min function used in Eq. (4.33) for the definition of the compressive flux ϕ_{rf}, namely:

$$\texttt{phic} \longleftrightarrow C_\alpha \frac{|\phi_f|}{|\mathbf{S}_f|}. \tag{4.36}$$

```
    // Standard face-flux compression coefficient
    surfaceScalarField phic(mixture.cAlpha()*mag(phi/mesh.magSf()));
```

We recall that the user can set a value for the coefficient C_α (denoted as `cAlpha`), otherwise a default value is considered. Next, the boundary conditions of `phic` are checked because there must not be interface compression at non-coupled boundary faces like inlets and outlets. This check is facilitated by the use of a pointer `phicBf` to the boundary faces of the `phic` field.

```
    surfaceScalarField::Boundary& phicBf =
        phic.boundaryFieldRef();

    // Do not compress interface at non-coupled boundary faces
    // (inlets, outlets etc.)
```

```
81      forAll(phic.boundaryField(), patchi)
82      {
83          fvsPatchScalarField& phicp = phicBf[patchi];
84
85          if (!phicp.coupled())
86          {
87              phicp == 0;
88          }
89      }
```

A new variable named `phiCN` is created to host the time-blended volumetric flux and it is initialized as the current volumetric flux `phi`. Since the volumetric flux is always computed explicitly, then its time blending is done over the previous time step (so we introduce another notation):

$$\begin{aligned} \text{If C-N:} \quad & \texttt{phiCN} \longleftrightarrow \phi^n_{f,CN} = \theta_{CN}\phi^n_f + (1-\theta_{CN})\phi^{n-1}_f \\ \text{If Euler:} \quad & \texttt{phiCN} \longleftrightarrow \phi^n_{f,CN} = \phi^n_f. \end{aligned}$$

```
91      tmp<surfaceScalarField> phiCN(phi);
92
93      // Calculate the Crank-Nicolson off-centred volumetric flux
94      if (ocCoeff > 0)
95      {
96          phiCN = cnCoeff*phi + (1.0 - cnCoeff)*phi.oldTime();
97      }
```

4.3.2 Predictor step

The predictor implicit step is computed when `MULESCorr = true` by solving the equation

$$\tilde{\alpha}^{n+1}_P = \alpha^n_P - \frac{\Delta t}{|V_P|}\sum_{f\in\partial V_P}\phi^n_{f,CN}\tilde{\alpha}^{n+1}_{f,upwind} + \texttt{Su} + \texttt{Sp}\cdot\tilde{\alpha}^{n+1}_P,$$

where `Su` and `Sp` refers to the source terms to be treated explicitly and implicitly respectively (and we eventually refer the reader to §1.3.2.1 and to Eq. (1.133) for the discretization of the source terms). In the case of the `interFoam` solver (and even for our `interThermalRadConvFoam` solver) the α equation does not have any source term, therefore the terms `Su`, `Sp` that are initialized to zero from the module `alphaSuSp.H` remain null. The linear system to solve is called `alpha1Eqn` because the two phases are referred respectively as

$$\begin{aligned} \texttt{alpha1} &\longleftrightarrow \alpha, \\ \texttt{alpha2} &\longleftrightarrow 1-\alpha. \end{aligned}$$

```
101         #include "alphaSuSp.H"
102
103         fvScalarMatrix alpha1Eqn
104         (
105             (
106                 LTS
107               ? fv::localEulerDdtScheme<scalar>(mesh).fvmDdt(alpha1)
108               : fv::EulerDdtScheme<scalar>(mesh).fvmDdt(alpha1)
```

```
109             )
110           + fv::gaussConvectionScheme<scalar>
111             (
112                 mesh,
113                 phiCN,
114                 upwind<scalar>(mesh, phiCN)
115             ).fvmDiv(phiCN, alpha1)
116         // - fvm::Sp(fvc::ddt(dimensionedScalar("1", dimless, 1), mesh)
117         //           + fvc::div(phiCN), alpha1)
118           ==
119             Su + fvm::Sp(Sp + divU, alpha1)
120         );
121
122         alpha1Eqn.solve();
```

After that, the implicit equation is solved and the predicted value $\widetilde{\alpha}_P^{n+1}$ is determined and stored in the `alpha1` field. Meanwhile, the flux calculated with the upwind scheme is stored inside a new variable called `alphaPhi10`. In our case, since the source terms of the α equation are null, we have simply:

$$\texttt{alphaPhi10} \longleftrightarrow \phi_{f,CN}^{n}\widetilde{\alpha}_{f,upwind}^{n+1} = \widetilde{F}_U^{n+1}. \tag{4.37}$$

We highlight that the surface scalar field `alphaPhi10` is a global variable because it is defined inside the module VoF/createAlphaFluxes.H called from the `interFoam` main function.

```
130         tmp<surfaceScalarField> talphaPhi1UD(alpha1Eqn.flux());
131         alphaPhi10 = talphaPhi1UD();
```

With the predicted field of the liquid phase fraction stored in `alpha1`, the gaseous phase fraction field called `alpha2` is updated and then also the fields of the transport properties are corrected, which means that the density field is updated according to the fields `alpha1` and `alpha2`, and the same is for the other fields associated to the transport properties of the two fluids, such as viscosity.

```
152         alpha2 = 1.0 - alpha1;
153
154         mixture.correct();
```

4.3.3 Compression MULES corrector iterations

After the predictor step, a for-cycle loop starts on the counter variable `aCorr` if the user (or the default setting) gives a value greater than zero to the parameter `nAlphaCorr` which is located in case/system/fvSolution. Every cycle begins by updating the source terms `Su` and `Sp` because they might depend on the predicted values and also on the corrected values, but in our case nothing changes because the source terms are null. After that, the compressive flux ϕ_{rf} (defined in Eq. (4.33)) is assembled by multiplying the compression coefficient `phic` (refer to Eq. 4.36) by the term n_f, which depends on the gradient of α according to Eq. (4.34).

```
158     for (int aCorr=0; aCorr<nAlphaCorr; aCorr++)
159     {
160         #include "alphaSuSp.H"
161
162         surfaceScalarField phir(phic*mixture.nHatf());
```

We underline that the compression coefficient `phic` has been initialized at the initial setting §4.3.1 so it is evaluated at the current time step t_n, whereas the term `nHatf`, that depends on the `alpha1` gradient, is evaluated at every cycle because initially the `alpha1` variable hosts the predicted value and then, after every for-cycle ν, it hosts the updated/corrected value.

$$\texttt{aCorr} \longleftrightarrow \nu,$$
$$\texttt{nHatf} \longleftrightarrow n_f^{(\nu-1)} = \frac{(\nabla\alpha^{(\nu-1)})_f}{|(\nabla\alpha^{(\nu-1)})_f + \delta_N|} \cdot \mathbf{S}_f.$$

As a consequence of what stated, the compressive volumetric flux at the ν-cycle is

$$\texttt{phir} \longleftrightarrow \phi_{rf}^{(n,\nu-1)} = C_\alpha \frac{|\phi_f^n|}{|\mathbf{S}_f|} n_f^{(\nu-1)}. \tag{4.38}$$

We specify that for $\nu = 0$, namely at the first iteration, $\alpha^{(\nu-1)}$ is considered as the predicted value $\widetilde{\alpha}^{n+1}$.

The high order flux $F_{f,H}$ defined in Eq. (4.32) is computed explicitly and stored inside the variable `talphaPhi1Un` according to the following equation

$$\begin{aligned}\texttt{talphaPhi1Un} \longleftrightarrow F_{f,H}^{(\nu-1)} &= \phi_{f,CN}^n \Big[\theta_{CN}\alpha_{f,div}^{(\nu-1)} + (1-\theta_{CN})\alpha_{f,div}^{(\nu-2)}\Big] \\ &\quad + \phi_{rf}^{(n,\nu-1)}\alpha_{rf}^{(\nu-1)}(1-\alpha)_{rf}^{(\nu-1)} \\ &= \phi_{f,CN}^n \alpha_{f,div,CN}^{(\nu-1)} + \phi_{rf}^{(n,\nu-1)}\alpha_{rf}^{(\nu-1)}(1-\alpha)_{rf}^{(\nu-1)}.\end{aligned} \tag{4.39}$$

We underline that the first term of the high order flux, that was called $\phi_f\alpha_{f,div}$, is now calculated with the Crank-Nicolson time blending (so we denoted it as $\alpha_{f,div,CN}^{(\nu-1)}$); at the first iteration, when `aCorr` $= 0 = \nu$, $\alpha_{f,div}^{(\nu-1)}$ coincides with the predicted value $\widetilde{\alpha}_{f,div}^{n+1}$ whereas $\alpha_{f,div}^{(\nu-2)}$ is the value $\alpha_{f,div}^n$.

```
164         tmp<surfaceScalarField> talphaPhi1Un
165         (
166             fvc::flux
167             (
168                 phiCN(),
169                 cnCoeff*alpha1 + (1.0 - cnCoeff)*alpha1.oldTime(),
170                 alphaScheme
171             )
172           + fvc::flux
173             (
174                -fvc::flux(-phir, alpha2, alpharScheme),
175                 alpha1,
176                 alpharScheme
177             )
178         );
```

MULES correction

When `MULESCorr = true` the anti-diffusive flux A_f is stored into a local variable `talphaPhi1Corr` and the current `alpha1` field is stored in a support variable.

$$\texttt{talphaPhi1Corr} \longleftrightarrow A_f^{(\nu-1)} = F_{f,H}^{(\nu-1)} - F_{f,U}^{(\nu-1)}.$$

After that, the function `MULES::correct()` is called to compute the limiter $\lambda_M^{(\nu)}$ and then update the variable `alpha1` according to the following equation:

$$\alpha_P^{(\nu)} = \alpha_P^{(\nu-1)} - \frac{\Delta t}{|V_P|} \sum_{f \in \partial V_P} \lambda_M^{(\nu)} A_f^{(\nu-1)}.$$

At the first cycle with $\nu = 0$, the flux $F_{f,U}^{(\nu-1)}$ is exactly the upwind flux evaluated with the predicted variable, Eq. (4.37); at the next cycles the flux $F_{f,U}^{(\nu-1)}$ hosts the corrected flux computed at the previous step. In the predicted step, the upwind flux can be interpreted as the first approximation of the corrected flux. According to this, the limitation of λ_M on the anti-diffusive flux concerns only the corrected flux instead of the total flux. We guess that the reason why this kind of computation has been associated with a variable named `MULESCorr` is because the MULES limiter is related to the corrected flux.

```
180         if (MULESCorr)
181         {
182             tmp<surfaceScalarField> talphaPhi1Corr(talphaPhi1Un() -
    alphaPhi10);
183             volScalarField alpha10("alpha10", alpha1);
184 
185             MULES::correct
186             (

195             );
```

The function `MULES::correct()` is implemented in MULES/CMULESTemplates.C (lines 139–191) and not reported here. This function modifies two variables: the anti-diffusive flux that is limited and whose value is stored again in `talphaPhi1Corr`, and the `alpha1` field

$$\texttt{talphaPhi1Corr} \longleftrightarrow \lambda_M^{(\nu)} A_f^{(\nu-1)},$$
$$\texttt{alpha1} \longleftrightarrow \alpha_P^{(\nu)}.$$

Inside the variable `alphaPhi10`, that originally stored the diffusive flux computed by the upwind scheme — see Eq. (4.37), the new anti-diffusive limited contribution is added, therefore the variable `alphaPhi10` hosts the corrected flux that will be used, at the next for-cycle, as the low order flux that needs to be compressed and corrected:

$$\begin{aligned} \texttt{alphaPhi10} \longleftrightarrow & F_{f,U}^{(\nu-1)} + \lambda_M^{(\nu)} A_f^{(\nu-1)} \\ & = F_{f,C}^{(\nu-1)} \\ & = F_{f,U}^{(\nu)}. \end{aligned} \tag{4.40}$$

If the current for-cycle is not the first one, then an under-relax evaluation is applied to the updated variables.

```
197             // Under-relax the correction for all but the 1st corrector
198             if (aCorr == 0)
199             {
200                 alphaPhi10 += talphaPhi1Corr();
201             }
202             else
```

```
203             {
204                 alpha1 = 0.5*alpha1 + 0.5*alpha10;
205                 alphaPhi10 += 0.5*talphaPhi1Corr();
206             }
```

Finally, the gaseous phase fraction field and the transport properties are updated (such as the density and viscosity fields).

```
225         alpha2 = 1.0 - alpha1;
226
227         mixture.correct();
```

4.3.4 Final assignments

When the for-cycle ends, namely when `aCorr`=`nAlphaCorr`, the fields `alpha1` and `alpha2` contain the new values of the phase fraction fields, and the global variable `alphaPhi10` stores the final corrected flux $F_{f,C}$. The last thing to compute is the mass flux due to the new phase fraction flux.

```
250         rhoPhi = alphaPhi10*(rho1f - rho2f) + phiCN*rho2f;
```

4.4 MULES limiter to explicit solution

When `MULESCorr = false` no predictor-corrector procedure is done, instead a simpler explicit iterative algorithm is performed. We go again over the `alphaEqn.H` file to comment on the implementation of such an algorithm.

The initial setting presented in §4.3.1 is the only part of the previous procedure valid even for this algorithm, therefore, before entering in the details of the iterations of the MULES scheme, we remark that we have the following variables initialized:

$$\texttt{phic} \longleftrightarrow C_\alpha \frac{|\phi_f^n|}{|\mathbf{S}_f|},$$
$$\texttt{phiCN} \longleftrightarrow \phi_{f,CN}^n = \theta_{CN}\phi_f^n + (1-\theta_{CN})\phi_f^{n-1};$$

that is, the compression coefficient and the volumetric flux blended with the Crank-Nicolson coefficient.

4.4.1 Compression MULES iterations

The for-cycle starts on the counter variable `aCorr` and lasts until it reaches the `nAlphaCorr` value. Initially, we find the same assignments presented before for the compressive flux `phir`, Eq. (4.38), and the high order flux `talphaPhi1Un`, Eq. (4.39):

$$\texttt{aCorr} \longleftrightarrow \nu,$$
$$\texttt{phir} = \texttt{phic} * \texttt{nHatf} \longleftrightarrow C_\alpha \frac{|\phi_f^n|}{|\mathbf{S}_f|} n_f^{(\nu-1)} = \phi_{rf}^{(n,\nu-1)},$$
$$\texttt{talphaPhi1Un} \longleftrightarrow F_{f,H}^{(\nu-1)} = \phi_{f,CN}^n \alpha_{f,div,CN}^{(\nu-1)} + \phi_{rf}^{(n,\nu-1)} \alpha_{rf}^{(\nu-1)} (1-\alpha)_{rf}^{(\nu-1)}.$$

In this case with `MULESCorr = false`, the global variable `alphaPhi10` hosts the high order flux `talphaPhi1Un` and the function `MULES::explicitSolve()` is called to find the limiter $\lambda_M^{(\nu)}$ that must be used in the following equation:

$$\begin{aligned}\alpha_P^{(\nu)} &= \alpha_P^{(\nu-1)} - \frac{\Delta t}{|V_P|}\sum_{f\in\partial V_P}\left[F_{f,U}^{(\nu-1)} + \lambda_M^{(\nu)} A_f^{(\nu-1)}\right] \\ &= \alpha_P^{(\nu-1)} - \frac{\Delta t}{|V_P|}\sum_{f\in\partial V_P}\left[F_{f,U}^{(\nu-1)} + \lambda_M^{(\nu)}\left(F_{f,H}^{(\nu-1)} - F_{f,U}^{(\nu-1)}\right)\right].\end{aligned} \tag{4.41}$$

```
208         else
209         {
210             alphaPhi10 = talphaPhi1Un;
211
212             MULES::explicitSolve
213             (
214                 geometricOneField(),
215                 alpha1,
216                 phiCN,
217                 alphaPhi10,
218                 Sp,
219                 (Su + divU*min(alpha1(), scalar(1)))(),
220                 oneField(),
221                 zeroField()
222             );
223         }
```

We remark that all the fluxes in Eq. (4.41) are evaluated with the current value $\alpha_P^{(\nu-1)}$; therefore, if the user chooses the Euler scheme for the time discretization of the α equation and the upwind scheme for `div(phi,alpha)`, then $\alpha_{f,div,CN}$ coincides with $\alpha_{f,upwind}^{(\nu-1)}$ and it descends a cancellation inside the anti-diffusive flux, which globally reduces to the compressive flux alone:

$$\begin{aligned}A_f^{(\nu-1)} &= F_{f,H}^{(\nu-1)} - F_{f,U}^{(\nu-1)} \\ &= \cancel{\phi_{f,CN}^{n}\alpha_{f,upwind}^{(\nu-1)}} + \phi_{rf}^{(n,\nu-1)}\alpha_{rf}^{(\nu-1)}(1-\alpha)_{rf}^{(\nu-1)} - \cancel{\phi_{f,CN}^{n}\alpha_{f,upwind}^{(\nu-1)}}.\end{aligned}$$

Moreover, if in this situation the user assigns `cAlpha`=0, the anti-diffusive flux is identically zero and hence the MULES iterations have no effect. In such case, the α-equation is solved by the upwind scheme.

The function `MULES::explicitSolve()` is implemented inside MULES/MULESTemplates.C, lines 153–181, and is not reported here. Within the `MULES::explicitSolve()` function, the missing fields are built, namely the low order flux, the anti-diffusive flux, and afterwards the corrected flux. Furthermore, this function modifies the fields `alpha1` and `alphaPhi10`: the first one has the updated values $\alpha_P^{(\nu)}$ that are used again to compute all the fluxes at the next cycle and to apply the new equation (4.41), meanwhile, the other modified variable, `alphaPhi10`, contains the corrected flux $F_{f,C}^{(\nu-1)}$, but this value is not directly used at the next iteration; on the contrary, the variable is immediately redefined.

After the call to the function `MULES::explicitSolve()`, both the gaseous phase fraction field and the transport properties (like viscosity and density) are updated.

```
225         alpha2 = 1.0 - alpha1;
226
227         mixture.correct();
```

When the for-cycle stops, the same final assignments described in §4.3.4 are executed, namely, the fields `alpha1` and `alpha2` store the new values of the liquid and gaseous fraction fields, and the global variable `alphaPhi10`, which contains the final corrected flux, is used to compute the mass flux `rhoPhi` that is caused by the new phase fraction flux.

```
250         rhoPhi = alphaPhi10*(rho1f - rho2f) + phiCN*rho2f;
```

4.5 interThermalRadConvFoam design and structure

The governing equations solved from our solver `interThermalRadConvFoam` are the following:

$$\nabla \cdot \mathbf{u} = 0$$

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot \left(\rho \mathbf{u}\mathbf{u}^T\right) = -\nabla p^* + \mathbf{g} \cdot \mathbf{x} \nabla \rho + \nabla \cdot \boldsymbol{\tau} + \mathbf{f}_\Sigma, \tag{4.42}$$

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\mathbf{u}\alpha) + \nabla \cdot [\mathbf{u}_r \alpha(1-\alpha)] = 0, \tag{4.43}$$

$$\frac{\partial(\rho c_p T)}{\partial t} + \nabla \cdot (\rho c_p T \mathbf{u}) - \bar{\chi}_\Sigma \Delta(kT) = -\frac{\epsilon \sigma_{SB} f A_{fs}}{Vol}\left(T^4 - T_{env}^4\right) - \frac{\lambda f A_{fs}}{Vol}\left(T - T_{env}\right).$$

The mass, momentum, and phase volumetric fraction equations are the same as in the original `interFoam` solver, whereas the energy equation is our contribution, and its implementation inside the software is described in a Tutorial reported in the Appendix §A.

The previous equations (actually solved) have little differences from those equations we derived in §2.2 (and recalled in Eqs. (4.1)) in the right-hand side of the momentum equation and in the α-equation. We notice that the differences in the momentum eq. (4.42) with respect to Eq. (4.1b) are referred to the pressure and gravitational terms. It is a common practice, when adopting the VOF method, to use a modified pressure p^* defined as follows

$$p^* := p - \rho g z = p - \rho \mathbf{g} \cdot \mathbf{x}, \tag{4.44}$$

because this is convenient when declaring and applying the boundary conditions for pressure. Such a modified pressure does not contain the hydrostatic component, which is the problematic part in the multiphase model since it takes into account density and therefore it requires particular attention as long as the fluids have two different values of that. In fact, for a system with a shared pressure field, the vertical component of pressure must be different for each phase because of the hydrostatic component ρg that depends on the densities of fluids. We show how the pressure and gravitational terms in the standard momentum Eq. (4.1b) changes because of the introduction of p^*:

$$\begin{aligned} -\nabla p + \rho \mathbf{g} &\overset{(4.44)}{=} -\nabla p^* - \nabla(\rho g z) + \rho \mathbf{g} \\ &= -\nabla p^* - g z \nabla \rho - \rho \nabla(g z) + \rho \mathbf{g} \\ &= -\nabla p^* - g z \nabla \rho - \cancel{\rho \mathbf{g}} + \cancel{\rho \mathbf{g}} \\ &= -\nabla p^* - g z \nabla \rho, \end{aligned}$$

where we assumed that the gravitational acceleration does not change vertically. We also notice that, since each fluid has constant density, $\nabla \rho = 0$ everywhere except at the phase interface.

Furthermore, the α-equation (4.43) presents an additional term with respect to Eq. (4.1c), i.e. $\nabla \cdot [\mathbf{u}_r \alpha(1-\alpha)]$, that is not accounted in the natural derivation obtained in §2.2.1.1. In fact, it is an artificial term active only on the phase interface, namely where $\alpha \neq 0, 1$. The velocity $\mathbf{u}_r$ that appears is called relative (or compression) velocity and we defined it in the framework of the Flux-Corrected Technique, in §4.2. Such a term acts to compress the interface between the phases, avoiding its spread over multiple cells.

Our solver `interThermalRadConvFoam` is constituted by several files gathered in a unique folder interThermalRadConvFoam located in applications/solvers/multiphase (the organization and structure of the OpenFOAM folders is described in §A.1). The directory interThermalRadConvFoam shares its location with other solvers for multiphase simulations (such as the original interFoam) and with the folder VoF that contains files for the implementation of the VOF method. Inside the interThermalRadConvFoam folder there are the following main files:

- interThermalRadConvFoam.C
- createFields.H
- alphaSuSp.H
- UEqn.H
- pEqn.H
- TEqn.H

The file interThermalRadConvFoam.C contains the main code of the solver which invokes the modules *.H. The module createFields.H initializes the fields, whereas the others UEqn.H, pEqn.H, and TEqn.H implement the momentum, pressure, and energy equation respectively. The main structure of the `interThermalRadConvFoam` solver is presented in the following Listing 4.2 (neglecting those lines that refer to topics that we do not deal with in the present work) and works as stated next:

- Initialize the fields p^*, $\mathbf{u}$, T, ϕ, α, ρ, c_p (among others), line 74.
- Set the initial time step Δt, lines 83-84.
- Start runtime loop, line 90.
  - Check the CFL condition to set the time step Δt, lines 100-102.
  - Advance in time, line 105.
  - Start the PIMPLE loop for outer iterations `nOuterCorrectors` times, line 110.
    - Solve the α-equation and update the fields α, ρ, and c_p, lines 151-152.
    - Solve the momentum equation, line 161.
    - Start the PISO loop for the inner iterations `nCorrectors` times, line 164.
      - Solve the Poisson equation for pressure, line 166.
  - Solve the energy equation, line 182.

```
26 Application
27     interThermalRadConvFoam
```

```
56 int main(int argc, char *argv[])
```

```
74     #include "createFields.H"
```

```
83         #include "CourantNo.H"
84         #include "setInitialDeltaT.H"
```

```
87      // * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
88      Info<< "\nStarting time loop\n" << endl;
89
90      while (runTime.run())
91      {

100             #include "CourantNo.H"
101             #include "alphaCourantNo.H"
102             #include "setDeltaT.H"

105         ++runTime;

109         // --- Pressure-velocity PIMPLE corrector loop
110         while (pimple.loop())
111         {

151             #include "alphaControls.H"
152             #include "alphaEqnSubCycle.H"

161             #include "UEqn.H"
162
163             // --- Pressure corrector loop
164             while (pimple.correct())
165             {
166                 #include "pEqn.H"
167             }

173         }
174
175         #include "TEqn.H"

180     }
181
182     Info<< "End\n" << endl;
183
184     return 0;
185 }
```

Listing 4.2: Main structure of `interThermalRadConvFoam.C` file.

4.6 Other solution schemes

As presented in the previous section §4.1, the ***segregated*** approach is a possibility for the numerical solution of a system of coupled PDEs; we have seen that the equations are solved one at time and the coupling is obtained by an iterative procedure, applying this technique in the *incompressible* limit (namely when the continuity equation reduces to a kinematic constraint).

The other option is the ***fully coupled*** approach: all the equations are discretized and linearized together by producing a single large system which is solved as a whole. In the *incompressible* limit, this approach was developed according with two strategies.

1. Without introducing a pressure equation, the momentum and continuity equations are discretized in a straightforward manner, see Caretto et al. [32], Braaten [26], Vanka [261], Karki and Mongia [140]. Because of the lack of a pressure equation, the main diagonal of the matrix associated to the discretized continuity equation

has zeros producing an ill-conditioned system of equations. This problem has been fixed in different ways, for example:

- by adopting a proper interpolation function that considers the neighboring pressure effect on the velocities, as done by Rhie and Chow [227] and Schneider and Raw [242],
- by passing through algebraic manipulations, as in Galpin and Raithby [96],
- by the penalty formulations, as proposed by Hanby and Silvester [113],
- by using the pre-conditioning technique, like Hanby et al. [114].

2. A pressure equation is derived and joins the momentum equation, see Patankar [207], Lonsdale [175]).

Since the numerical solution refers to the computational domain, the number of equations to solve depends on the number of the cells, as well as on the number of the constitutive equations that form the PDE system. So, despite the advantage of the fully coupled approach that manages to preserve the coupling between the unknowns, the amount of memory necessary for storage and the computational capabilities have been important limitations to its diffusion and evolution.

Instead, the segregated methods require the solution of systems smaller than those solved by the coupled methods, being referred to only one variable. In addition, is one decides to enrich a model by adding one more differential equation, in the segregated approach the new equation is treated separately so that the size of the whole algebraic system linearly grows; on the other hand, in the fully coupled approach the size growth is quadratical.

For these reasons, the segregated methods had the biggest development and diffusion for more than thirty years. In the last years, the enormous increment of computational capabilities finally allowed extensive use and progress even of fully coupled algorithms, [79].

Both methods cope with non-linear systems with iterations. The fully coupled algorithm might converge more robustly and in fewer iterations with respect to the segregated approach because of its intrinsically coupled nature. Anyway, each iterative step might require more memory and time for the computations, for these reasons the segregated methods can be faster overall. However, in the CFD community and in literature the debate about the performance of both types of methods is still open, and nowadays many CFD programs, such as OpenFOAM [137], ANSYS® FLUENT, and COMSOL® Multiphysics, have both approaches implemented and the user can switch between them.

4.7 Numerical simulations

In this section we present some preliminary results of numerical tests performed with `interThermalRadConvFoam`. These tests are based on two of the benchmarks proposed in [52] for lava flow simulations (BM1 and BM3) and already presented for the depth-averaged case in §3.4, hence we will not report all their details here. We remind that, differently from the depth-averaged case, where only the fluid is modeled, here we have to simulate the full 3D domain, and thus both the fluid of interest and the surrounding air. In any simulation presented, the air was modeled as a Newtonian fluid with density $\rho = 1\,\mathrm{kg\,m^{-3}}$ and kinematic viscosity $\nu = 1.48 \cdot 10^{-5}\,\mathrm{m^2 s^{-1}}$. Furthermore, the regime was

set to laminar (no turbulence effects are modeled), and the solver run in PISO mode with 3 inner loops (`nCorrectors 3`). Moreover, we underline that in the presented tests we assume constant viscosity, and further analysis on the temperature-dependent viscosity will be object of the future studies.

In addition, we present simulations executed with the use of a dynamic mesh with adaptive refinement, which means that the mesh may change during computation and that it might be locally refined according to some parameters given by the user. The dynamic and adaptive mesh refinement permits to select the portion of the domain to refine dynamically, increasing the level of discretization where the simulated processes require greater accuracy. In our case, we choose to refine those cells that host the phase interface, in particular, the cells with $0.01 \leqslant \alpha \leqslant 0.99$ because a good description of it is of paramount importance. The user can set the conditions for the dynamic and adaptive refinement inside the dictionary dynamicMeshDict located inside the folder case/constant related to the test case of interest.

In OpenFOAM the starting grid is hexahedral and each level of refinement divides each original cell into 8 subcells. Thus, a cell of the original coarser grid with n levels of refinement results in 8^n subcells. Besides, since the CFL condition (presented in §1.2.2.2) depends on the size of the spatial discretization, the use of the refinement leads to a reduction of the time step determined by the CFL condition.

4.7.1 BM1: Dam break of viscous fluids over a flat bottom

The first test is a classic dam-break simulation that starts with a fluid initially confined in a box and then, after the abrupt removal of one box side, it starts to flow into a channel. We described in details this benchmark in §3.4.2, so we recall here only some necessary information. For this test, we consider a Newtonian and isothermal fluid (with kinematic viscosity $\nu = 3.7\,\mathrm{m^2\,s^{-1}}$ and density $\rho = 2700\,\mathrm{kg\,m^{-3}}$), initially confined in a region with length 6.6 m and height 1 m. The 3D computational domain has a total length 72.6 m, total height 3.3 m, and total width 6.6 m. We remark again that, with respect to the depth-averaged simulations, here the specification of the domain height is required, and the atmospheric air surrounding the fluid of interest is also simulated by the model.

In this test, we focused on the capability of the model to reproduce a sharp interface between the fluid and the surrounding atmosphere. We adopted both static meshes and meshes with dynamic refinements and compared the results both in terms of computation time and in terms of the quality of the outputs obtained. The aspect ratio $\Delta x/\Delta z$ (where we are assuming that x refers to the horizontal orthogonal direction and z to the vertical direction) has been chosen as close as possible to unity. Otherwise, there is the risk that the free surface of the fluid is badly described on the horizontal-upper side if $\Delta x \ll \Delta z$, or at the front position if $\Delta x \gg \Delta z$.

First of all, in Figures 4.2, 4.3, 4.4, we show the three dimensional representation of the liquid free surface (represented by a contour of α) at $t = 10$, 100, 500 s respectively, for a simulation using a dynamic and adaptive mesh with 3 levels of refinement. These figures clearly show the grid refinement in the regions with large gradients of flow variables (as at the liquid/gas interface) and a 2D nature of the dynamics of flow propagation in the sense that the fluid behavior is invariant with respect to the y axis (this is consistent with the specifics of the benchmark BM1). Consequently, thanks to the 2D nature of the dynamics, we study and compare the performances of simulations computed with a static uniform 2D grid with those obtained with the dynamic mesh refinement over a 3D grid

(of which we will show only a cross-section of the 3D computational domain on the x-z plane in the following plots).

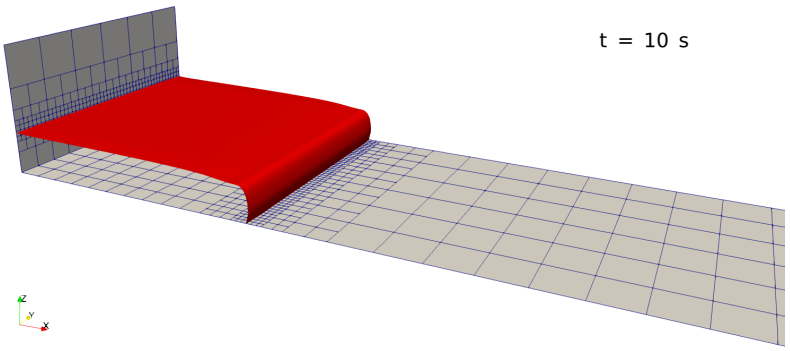


Figure 4.2: *Dam break of a viscous fluid over a flat bottom.* Free surface of the liquid phase. Simulation at $t = 10\,\mathrm{s}$ obtained using dynamic mesh refinement with 3 levels of refinement.

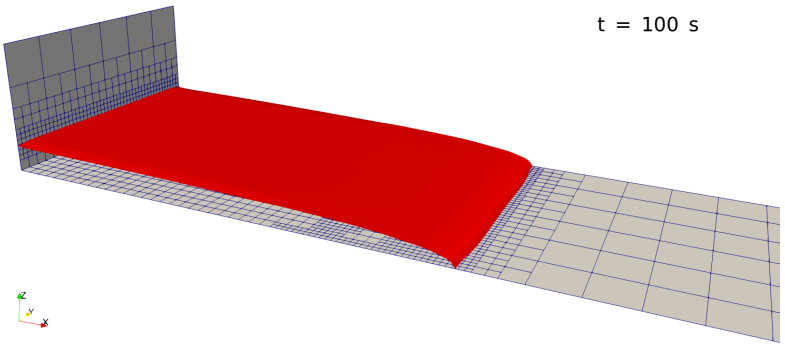


Figure 4.3: *Dam break of a viscous fluid over a flat bottom.* Free surface of the liquid phase. Simulation at $t = 100\,\mathrm{s}$ obtained using dynamic mesh refinement with 3 levels of refinement.

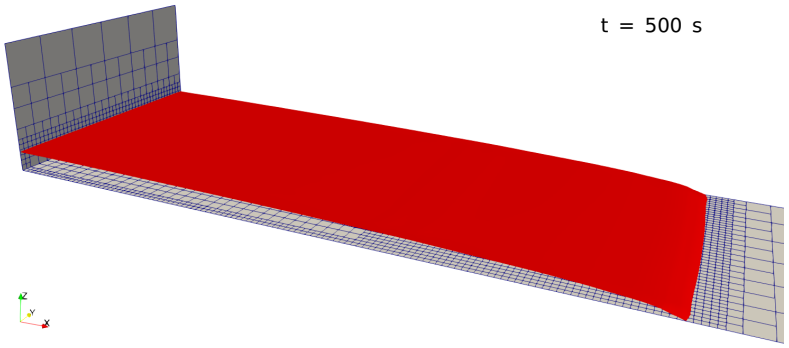


Figure 4.4: *Dam break of a viscous fluid over a flat bottom.* Free surface of the liquid phase. Simulation at $t = 500\,\mathrm{s}$ obtained using dynamic mesh refinement with 3 levels of refinement.

The first part of our analysis is on the use of the static 2D mesh with uniform discretization, Figures 4.5, 4.6, and 4.7 represent results obtained at $t = 10, 100, 500\,\mathrm{s}$ respectively. In each Figure, simulations obtained with four different grid resolutions are compared. The coarsest mesh was obtained with $\Delta x_C = \Delta z_C = 1.1\,\mathrm{m}$, and the others with $\Delta x = \Delta z = 0.55, 0.275, 0.1375\,\mathrm{m}$ (corresponding to $\Delta x_C/2$, $\Delta x_C/4$, and $\Delta x_C/8$) respectively.

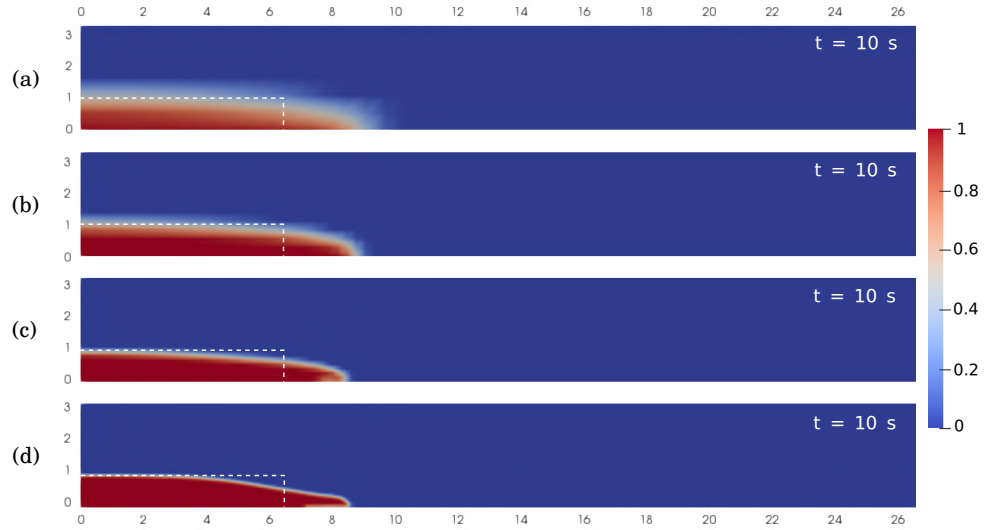


Figure 4.5: *Dam break of a viscous fluid over a flat bottom.* Comparison of results obtained using a static mesh and uniform 2D discretization. α-field at $t = 10\,\mathrm{s}$ (liquid: $\alpha = 1$, air: $\alpha = 0$). The dashed lines represent the initial condition. (a) cell size $\Delta x_C = 1.1\,\mathrm{m}$, (b) cell size $\Delta x_C/2$, (c) cell size $\Delta x/4$, (d) cell size $\Delta x_C/8$.

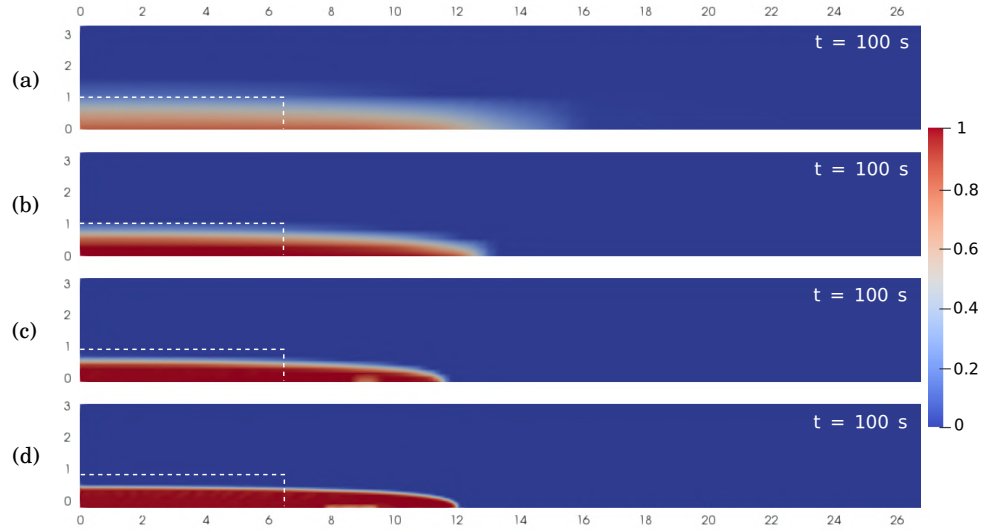


Figure 4.6: *Dam break of a viscous fluid over a flat bottom.* Comparison of results obtained using a static mesh and uniform 2D discretization. α-field at $t = 100\,\mathrm{s}$ (liquid: $\alpha = 1$, air: $\alpha = 0$). The dashed lines represent the initial condition. (a) cell size $\Delta x_C = 1.1\,\mathrm{m}$, (b) cell size $\Delta x_C/2$, (c) cell size $\Delta x/4$, (d) cell size $\Delta x_C/8$.

We analyze the results obtained by the simulations with a focus on the interface. The simulations obtained with the coarsest discretization grid (panels (a) of Figures 4.5, 4.6, 4.7) show a high diffusion of the interface, especially in correspondence of the front. In fact, even though at $t = 10\,\mathrm{s}$ the front presents a quite good agreement with those of the simulations obtained with finer meshes and at $t = 100\,\mathrm{s}$ the result is still not too far from others (even though the interface at the front is smeared almost over 8 cells, between $\sim 12\,\mathrm{m}$ and $\sim 16\,\mathrm{m}$), the description of the fluid front is completely lost at $t = 500\,\mathrm{s}$. The results obtained with $\Delta x_C/2$ (panels (b) of Figures 4.5, 4.6, 4.7) show that the description of the fluid interface suffers of less diffusion with respect to the case of the coarsest mesh

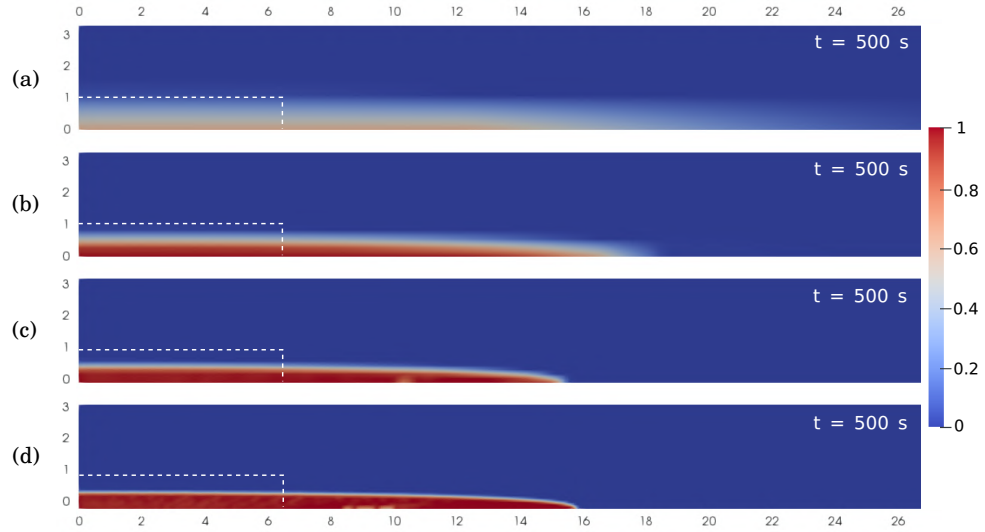


Figure 4.7: *Dam break of a viscous fluid over a flat bottom.* Comparison of results obtained using a static mesh and uniform 2D discretization. α-field at $t = 500\,\mathrm{s}$ (liquid: $\alpha = 1$, air: $\alpha = 0$). The dashed lines represent the initial condition. (a) cell size $\Delta x_C = 1.1\,\mathrm{m}$, (b) cell size $\Delta x_C/2$, (c) cell size $\Delta x/4$, (d) cell size $\Delta x_C/8$.

previously examined, so that at $t = 500\,\mathrm{s}$ the interface occupies almost 10 cells, between $\sim 15.5\,\mathrm{m}$ and $\sim 18.5\,\mathrm{m}$ (see also Figure 4.8, panel (b)). In addition, this makes it difficult to identify the exact position of the front, which also remains over-estimated compared to other simulations. Simulations computed with $\Delta x_C/4$ and $\Delta x_C/8$ (panels (c) and (d) of Figures 4.5, 4.6, 4.7) show comparable results and both describe the interface between fluid and air sharply. The interface at the front at time $t = 500\,\mathrm{s}$ is constituted by 3 cells and 1 cell respectively for these two simulations (see also Figure 4.8, panels (c) and (d)).

Lastly, we see from Figure 4.8 that the upper horizontal interface is described by just one layer of cells, hence the case with the finest discretization grid is the only one that succeeded in keeping the interface sharp even at the propagation front (the simulation with the coarsest mesh is not considered because at time $t = 500\,\mathrm{s}$ the solution is not reliable).

The computational time required for simulations of 500 s are reported in Table 4.1. All these simulations are serial runs (not parallel) on a single core. Theoretically, the computational time should increase by a factor 8 when the cell size is decreased by a factor 2. This is because for a 2D simulation the number of computational cells increase of a factor 4 and the time step decrease by a factor 2 (in accordance with the CFL condition). Here, we see that the computational time increases by a factor with is substantially smaller than 8, probably due to a smaller number of internal iterations required to reach the convergence in a single time step, when the time step decreases.

Δx	cells	time
1.1 m	66×6	3.39 s
0.55 m	132×12	7.69 s
0.275 m	264×24	25.72 s
0.1375 m	528×48	139.08 s

Table 4.1: *Dam break of a viscous fluid over a flat bottom.* Elapsed execution time for the simulations of 500 s over different discretization grids.

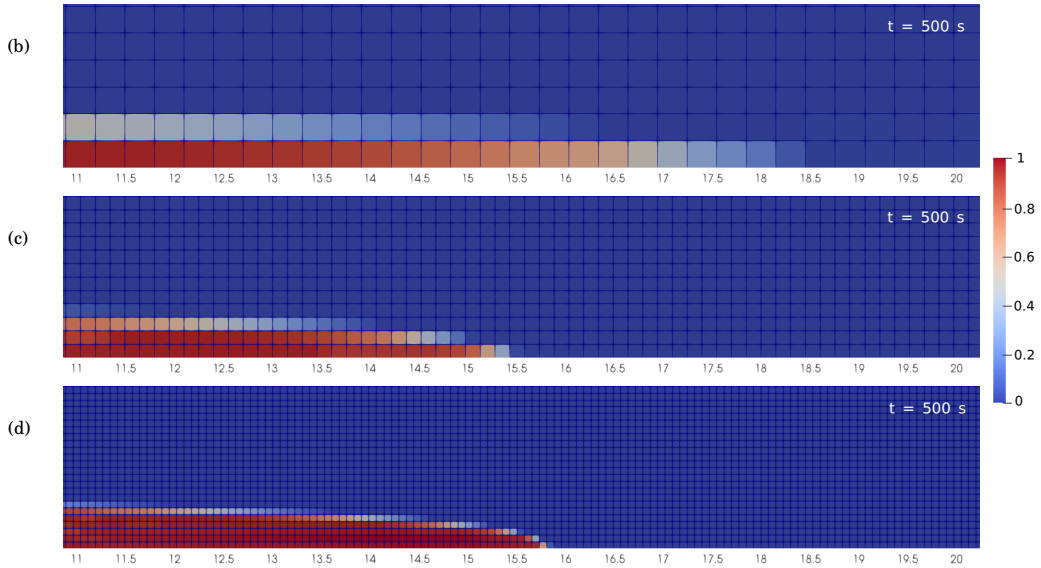


Figure 4.8: *Dam break of a viscous fluid over a flat bottom.* Comparison of the description of the front zone with results obtained using a static mesh and uniform 2D discretization. α-field at $t = 500\,\mathrm{s}$ (liquid: $\alpha = 1$, air: $\alpha = 0$). (b) cell size $\Delta x_C/2$, (c) cell size $\Delta x/4$, (d) cell size $\Delta x_C/8$ (with $\Delta x = 1.1\,\mathrm{m}$).

The second part of our analysis uses the dynamic mesh refinement of the cells close to the interface between air and liquid on a 3D mesh. The domain was initially discretized with a uniform 3D mesh made of $66 \times 6 \times 3$ cells, and we produced results with 2, 3, and 4 levels of refinement. Since having a cell aspect ratio close to one is often a necessity, we adopted the same length for the discretization step along the three directions. It is worth also underlying that the mesh refinement, in OpenFOAM, consists of dividing a cell into 8 sub-cells, therefore such a procedure always produces a 3D grid, even if the initial one was not. Figures 4.9, 4.10, 4.11 report the results at $t = 10$, 100, 500 s respectively and in Figure 4.9 (on the right panels), we highlighted in addition the edges of the meshes in order to appreciate the adaptive refinement.

By comparing Figures 4.9, 4.10, 4.11, that show simulations obtained using the adaptive 3D mesh refinement, with Figures 4.5, 4.6, 4.7, that report simulations computed over 2D static and uniform meshes, we notice a correspondence between them. Results obtained using the dynamic mesh with 2 refinement levels are similar to those computed over the uniform mesh with $\Delta x = \Delta x_C/4$; the simulation resulting from the use of the dynamic mesh with 3 refinement levels is comparable with that obtained with the uniform mesh $\Delta x = \Delta x_C/8$; finally, the outputs computed with the dynamic mesh with 4 refinement levels are very similar to those obtained using the finest uniform mesh with $\Delta x = \Delta x_C/16$. Despite the comparable descriptions of the fluid runout and of the phase interface, there is a huge gap between the simulations computed over a 2D mesh and those obtained with the 3D adaptive mesh refinement that is the execution time to run the code. In fact, whereas the 2D simulations are very fast to run (see Table 4.1), the 3D codes are definitely slower, see Table 4.2 that reports the execution times for the 3D cases.

We notice, for comparison, that the 2D simulation computed over the finest mesh grid took 139.08 s and that the same time 139.47 s was necessary for the case of adaptive mesh with 2 levels of refinement. However, we lastly stress over the great advantage

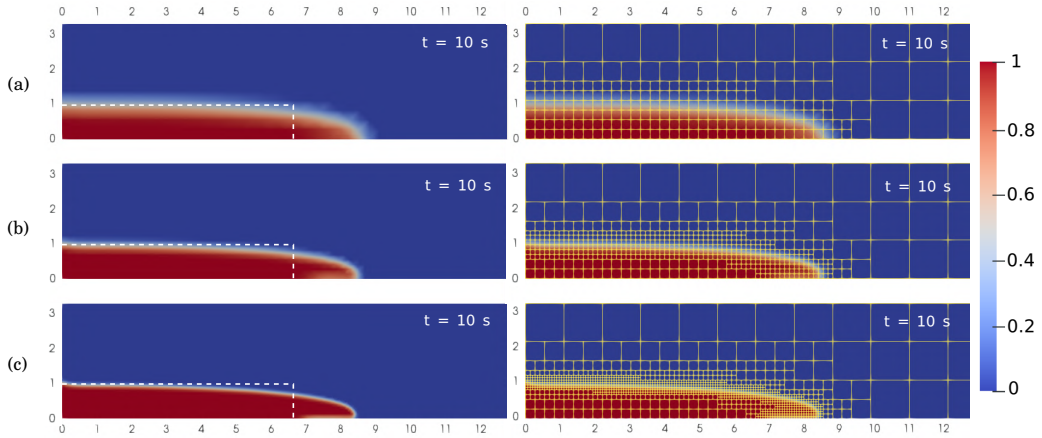


Figure 4.9: *Dam break of a viscous fluid over a flat bottom.* Comparison of results obtained using dynamic mesh refinement and different levels of refinement; the mesh was initially composed of $66 \times 6 \times 3$ cells. α-field is represented at $t = 10\,\mathrm{s}$ (liquid: $\alpha = 1$, air: $\alpha = 0$). The dashed lines represent the initial condition. (a) 2 levels of refinement, (b) 3 levels of refinement (c) 4 levels of refinement. The *right panels* highlight the discretization grid refinement.

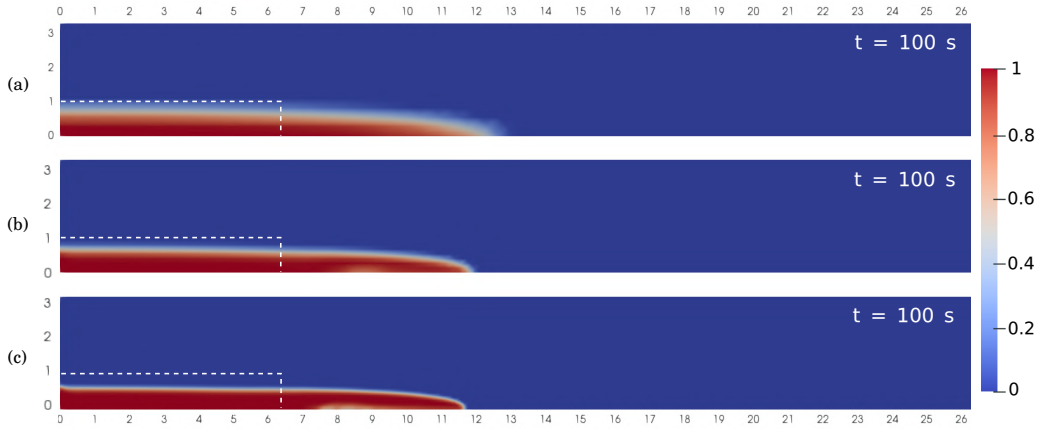


Figure 4.10: *Dam break of a viscous fluid over a flat bottom.* Comparison of results obtained using dynamic mesh refinement and different levels of refinement; the mesh was initially composed of $66 \times 6 \times 3$ cells. α-field is represented at $t = 100\,\mathrm{s}$ (liquid: $\alpha = 1$, air: $\alpha = 0$). The dashed lines represent the initial condition. (a) 2 levels of refinement, (b) 3 levels of refinement (c) 4 levels of refinement.

ref. level	time
2	139.47 s
3	1695.45 s
4	15367.60 s

Table 4.2: *Dam break of a viscous fluid over a flat bottom.* Elapsed execution time for the simulations of 500 s computed with the dynamic mesh refinement of the discretization grid which is initially composed of $66 \times 6 \times 3$ cells.

offered by the use of the dynamic refinement with respect to the use of a static 3D mesh. The simulation executed using the adaptive mesh with 3 refinement levels (where the smallest cell size is $\Delta x = 0.1375\,\mathrm{m}$) took 1695.45 s, whereas, a simulation obtained by using uniformly $\Delta x = 0.1375\,\mathrm{m}$ (producing a mesh with $528 \times 48 \times 24$ cells) required

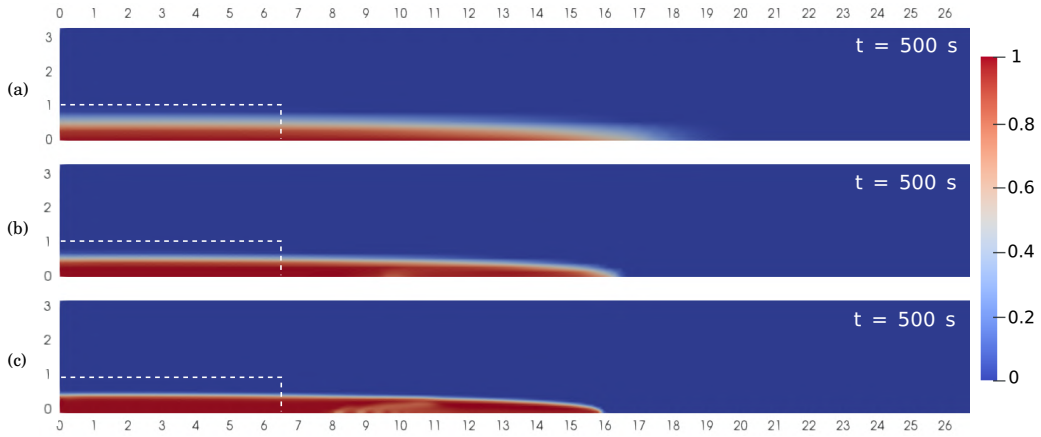


Figure 4.11: *Dam break of a viscous fluid over a flat bottom.* Comparison of results obtained using dynamic mesh refinement and different levels of refinement; the mesh was initially composed of $66 \times 6 \times 3$ cells. α-field is represented at $t = 500\,\text{s}$ (liquid: $\alpha = 1$, air: $\alpha = 0$). The dashed lines represent the initial condition. (a) 2 levels of refinement, (b) 3 levels of refinement (c) 4 levels of refinement.

twice the time, i.e. 3367.54 s. In addition, we depict in Figure 4.12 the results obtained using the uniform, static 3D mesh (represented in Figure 4.13) which are definitely equal to those obtained using the adaptive refinement shown in Figures 4.9, 4.10, 4.11 (panels (b)). Therefore, we can conclude that for this test the use of the adaptive refinement technique allows reducing the computational time by a factor two, with respect to the time needed to obtain a solution with a comparable accuracy but using a uniform mesh. The advantage of the mesh refinement is not only in the computational time but also in the amount of data to save for each output, because of the reduced total number of cells: in fact, whereas the uniform and static mesh has 608256 cells, the mesh in the dynamic case has in total 43656 cells at time $t = 10\,\text{s}$, 54810 cells at time $t = 100\,\text{s}$, and 64436 cells at time $t = 500\,\text{s}$, hence has at most one tenth of the cells of the uniform case. We remark that for large simulations also the amount of data saved (or the amount of data to transfer from one computer to another) could potentially put a constraint on the size of the simulation.

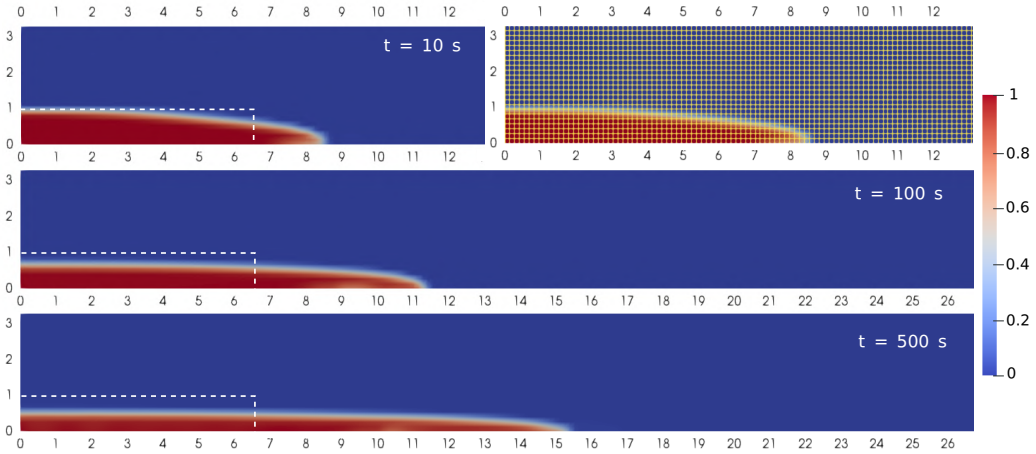


Figure 4.12: *Dam break of a viscous fluid over a flat bottom.* Simulation computed with a uniform, static 3D mesh with $\Delta x = 0.1375\,\text{m}$. α-field is represented at time $t = 10, 100, 500\,\text{s}$ (liquid: $\alpha = 1$, air: $\alpha = 0$). The dashed lines represent the initial condition.

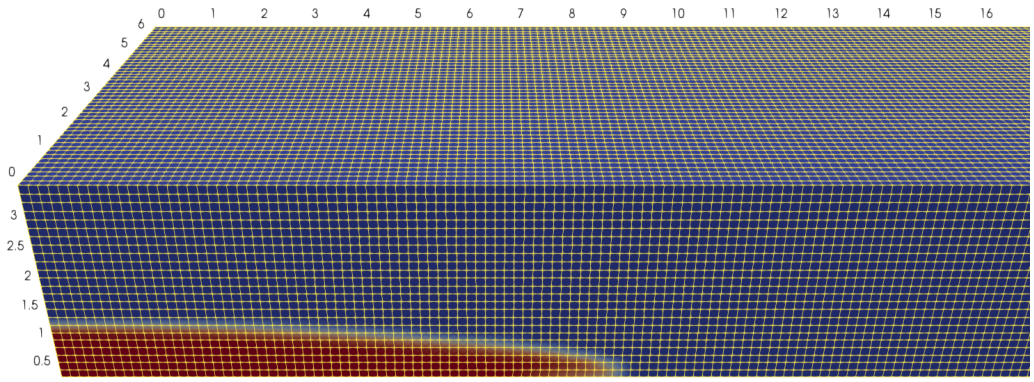


Figure 4.13: *Dam break of a viscous fluid over a flat bottom.* Uniform, static 3D mesh with $\Delta x = 0.1375\,\mathrm{m}$. α-field represented at time $t = 10\,\mathrm{s}$.

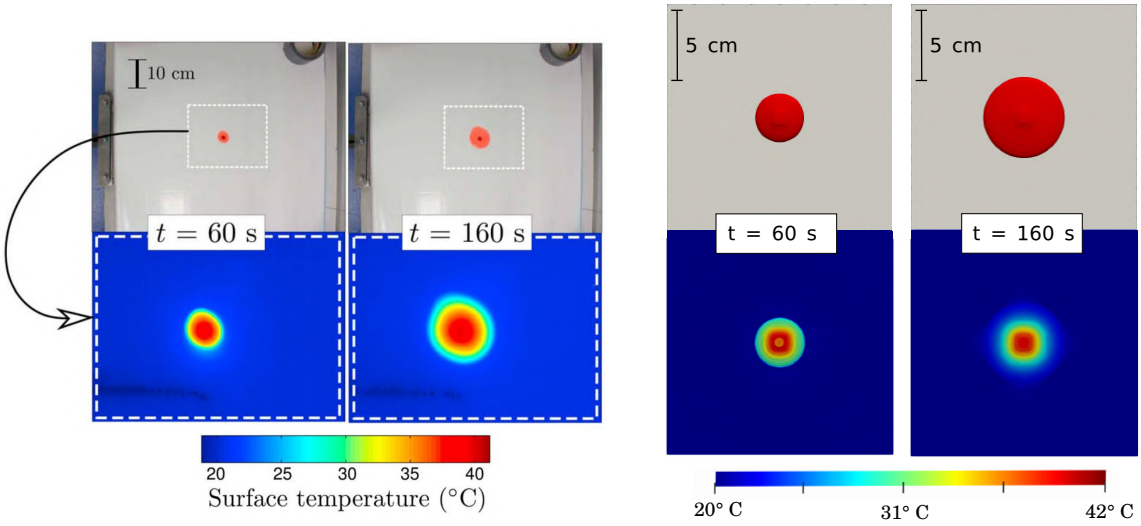
4.7.2 BM3: Axisymmetric cooling and spreading

The second test is the simulation of a hot viscous fluid injected on a horizontal plane that starts spreading and cooling in an axisymmetric fashion. This benchmark was already described in §3.4.6, hence here we remind only some necessary information. For this test we consider a Newtonian fluid (with kinematic viscosity $\nu = 3.56 \cdot 10^{-3}\,\mathrm{m^2\,s^{-1}}$ and density $\rho = 954\,\mathrm{kg\,m^{-3}}$) injected onto the plane with an effusion rate of $2.2 \cdot 10^{-8}\,\mathrm{m^3\,s^{-1}}$ from a hole of $4 \cdot 10^{-3}\,\mathrm{m}$ radius at the temperature $T_{vent} = 42°\mathrm{C}$ while the environment is at $T_{vent} = 20°\mathrm{C}$ (the complete set of physical parameters is reported in Table 3.5). The 3D computational domain that we adopted is $16 \cdot 10^{-2}\,\mathrm{m}$ long and wide, and only $10^{-2}\,\mathrm{m}$ high because the fluid propagates in a very thin layer. Once again, we remind that the air surrounding the fluid of interest is also simulated by the model, and that, with respect to the depth-averaged model, here the specification of the domain height is also necessary.

The main aim of the suite of simulations performed for this test is to study the performance obtained by using the adaptive mesh refinement (that helps to keep a sharp interface and to reduce the computation time with respect to a uniform, static, fine mesh). In addition, we discuss the scalability performance of the parallel execution, and, finally, we test the capability of the terms implemented in the energy equation of the model to properly describe the fluid heat loss through radiation, convection, and conduction.

Before a more quantitative analysis of our results, and reminding that this test has its origin in an analog laboratory experiment described in Garel et al. [99], we show a qualitative comparison between the results of our simulations with those of the experiment putting them alongside in Figure 4.14. Even though the pictures that represent the experiment and the simulation are not directly comparable, we appreciate that both of them show that the temperature gradient, from the center to the front of propagation, is more diffused at time $t = 160\,\mathrm{s}$ with respect to time $t = 60\,\mathrm{s}$.

Whereas in the previous figure we presented a 2D view from the top of simulation results, the simulations are fully 3D, and in Figures 4.15, 4.16, 4.17, 4.18, we present a three dimensional view of the liquid free surface (represent by a contour of α) at time $t = 20$, 60, 160, 380 s respectively, obtained from a simulation that uses an adaptive mesh with up to 4 levels of refinement, starting from a uniform grid with $\Delta x = 0.005\,\mathrm{m}$. The 3D plots clearly show that the grid refinement dynamically follows the liquid/air interface. A specification is necessary: because of the very small thickness of the surface, in Figures



(a) Picture taken from Garel et al. [99]. (b) Results of our simulation.

Figure 4.14: *Axisymmetric cooling and spreading.* (a) Optical (*top row*) and infrared (*bottom row*) images taken during an analog laboratory experiment at time $t = 60, 160\,\mathrm{s}$. The dashed rectangle in the optical image corresponds to the field of view of the infrared image below. (b) Top views of the fluid free-surface, represented as a contour of α (*top row*), and of the temperature (*bottom row*) at time $t = 60, 160\,\mathrm{s}$. The figures depict the top view of the whole domain $[-8; 8]$ cm $\times [-8; 8]$ cm. *Please note that the domain size represented in the two pictures are different.*

4.15, 4.16, 4.17, 4.18 we used a vertical scale increased by a factor 3, in order to better appreciate the three-dimensional nature of the flow surface.

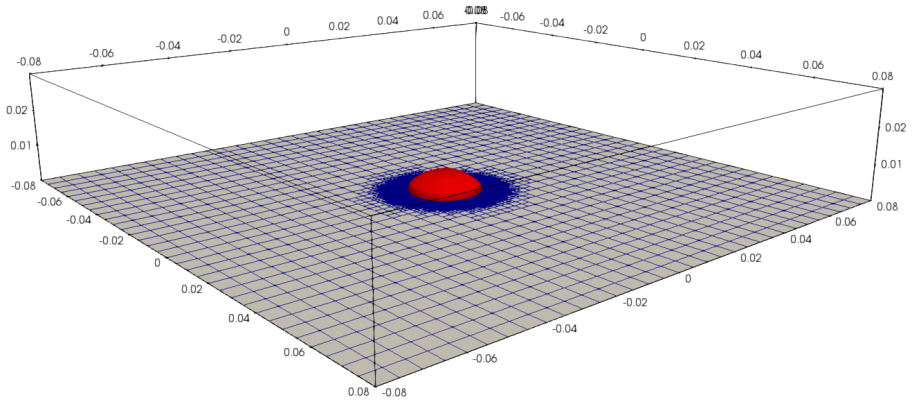


Figure 4.15: *Axisymmetric cooling and spreading.* Free-surface of the fluid phase, grid refinement of the horizontal plane, and computational domain outline. Simulation at time $t = 20\,\mathrm{s}$ obtained with adaptive mesh refinement up to 4 levels of refinement.

In Figures 4.19, 4.20, 4.21, we show and compare results and mesh refinements for simulations using different computational grids, at two times $t = 20, 380\,\mathrm{s}$. Figures 4.19, 4.20 show the results of simulations obtained with adaptive mesh that use up to 3 and 4 levels of refinement respectively, but that initially started with the same uniform mesh discretized with a grid resolution of $5 \cdot 10^{-3}\,\mathrm{m}$. As one might expect, the interface is described more sharply in the simulation with the highest level of refinement, i.e. 4, represented in Figure 4.20. Figure 4.21 depicts results of a simulation computed with the

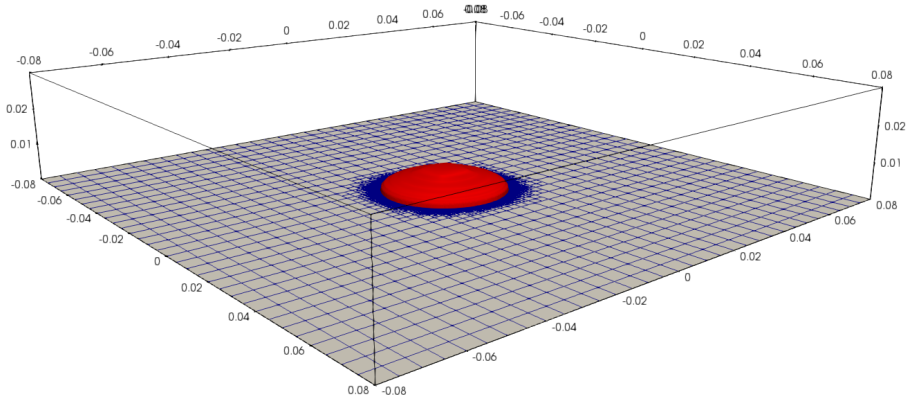


Figure 4.16: *Axisymmetric cooling and spreading.* Free-surface of the fluid phase, grid refinement of the horizontal plane, and computational domain outline. Simulation at time $t = 60\,\mathrm{s}$ obtained with dynamic mesh refinement up to 4 levels of refinement.

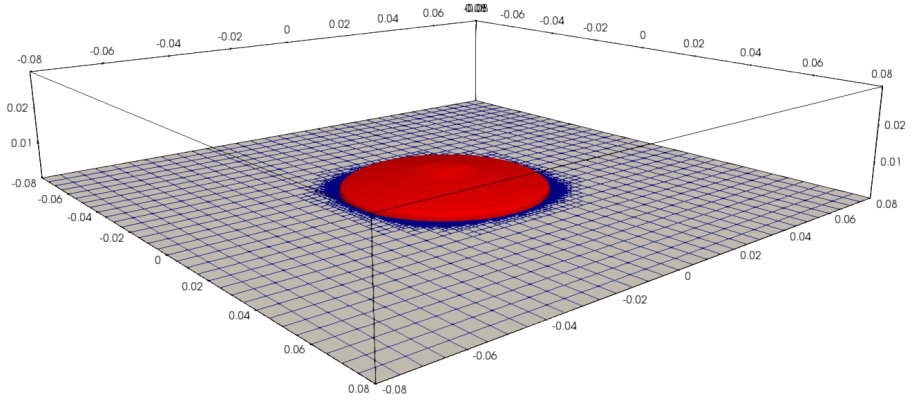


Figure 4.17: *Axisymmetric cooling and spreading.* Free-surface of the fluid phase, grid refinement of the horizontal plane and computational domain outline. Simulation at time $t = 160\,\mathrm{s}$ obtained with dynamic mesh refinement up to 4 levels of refinement.

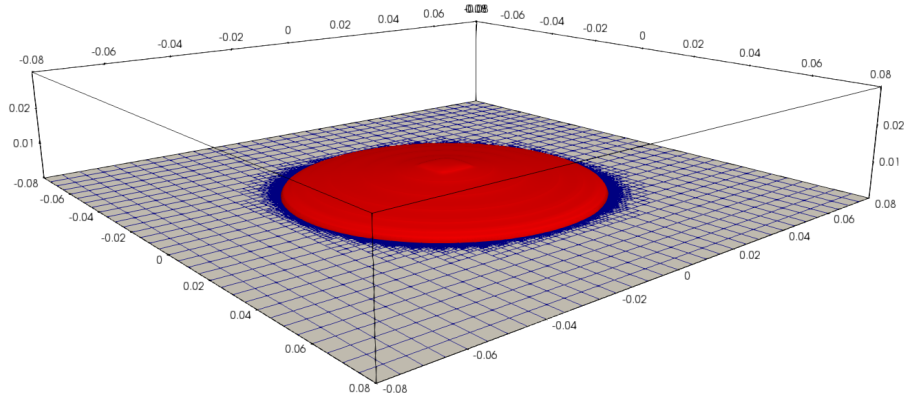


Figure 4.18: *Axisymmetric cooling and spreading.* Free-surface of the fluid phase, grid refinement of the horizontal plane, and computational domain outline. Simulation at time $t = 380\,\mathrm{s}$ obtained with dynamic mesh refinement up to 4 levels of refinement.

use of dynamic refinement up to level 3 and with an initial uniform mesh discretized with a grid resolution of $2.5 \cdot 10^{-3}\,\mathrm{m}$ (half the resolution of the previous simulations). The results of this last case are comparable to those obtained with an initial grid resolution of $5 \cdot 10^{-3}\,\mathrm{m}$ and 4 refinement levels, in terms of sharpness of the phase interface, and

execution time of the code (see Table 4.3).

initial Δx	ref. level	time
0.005 m	3	8112.01 s
0.005 m	4	31486.70 s
0.0025 m	3	32348.10 s

Table 4.3: *Axisymmetric cooling and spreading.* Elapsed execution time for the simulations of 380 s with different mesh refinements.

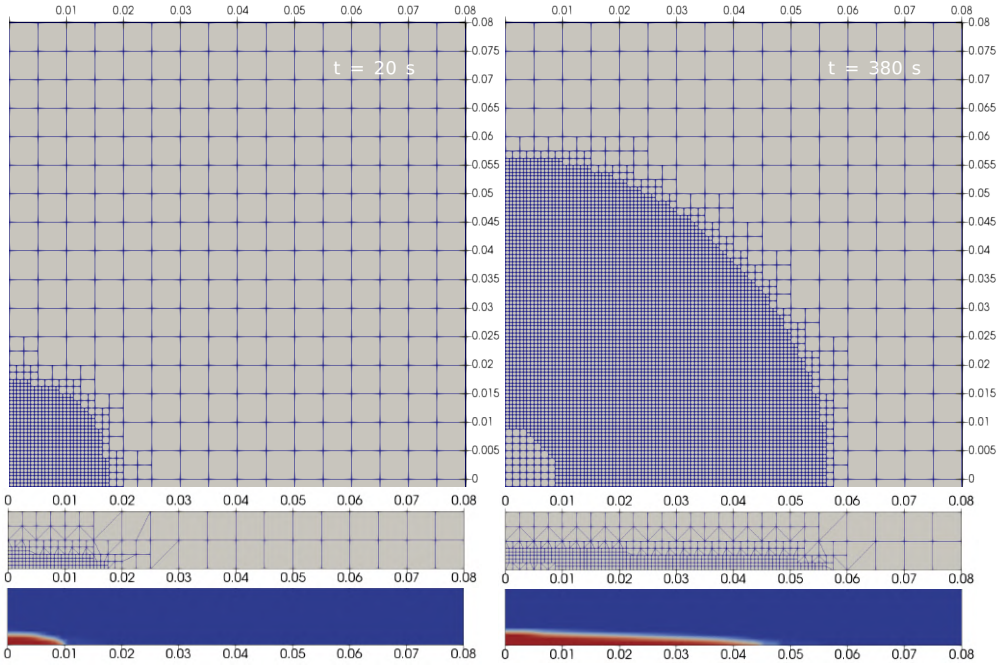


Figure 4.19: *Axisymmetric cooling and spreading.* Simulation computed with the dynamic mesh refinement up to 3 refinement levels over an initial uniform mesh defined by $\Delta x = 0.005$ m represented at time $t = 20, 380$ s. *Top row:* bottom view of the discretization grid. *Middle row:* side view of the discretization grid on the plane $x = 0$. *Bottom row:* side view of the α-field on the plane $x = 0$.

In every case depicted in Figures 4.19, 4.20, 4.21, we appreciate both the refinement of the meshes (which occurs as phase interface propagates), and the subsequent mesh coarsening. Furthermore, we also notice that in Figure 4.21 there is a refinement in correspondence of the origin that does not coarse. This is caused by the fact that, for this simulation, we used a finer initial grid in correspondence of the inlet hole, namely centered in $(0, 0, 0)$ and with radius $4 \cdot 10^{-3}$ m, in order to better capture the circular shape of the hole. By comparing the results, we see that such imposed refinement does not affect the results.

In addition, we have further investigated the effect of the grid size, by comparing a simulation using a uniform and static mesh with a grid resolution of $6.25 \cdot 10^{-4}$ m, corresponding to the minimum cell size obtained using a dynamic mesh with an initial grid resolution of $5 \cdot 10^{-3}$ m and 3 refinement levels. From Figure 4.22 we can see that results of the two simulations are very similar, both in terms of runout and sharpness of the interface. It is important to notice that in this case also the computational time required is

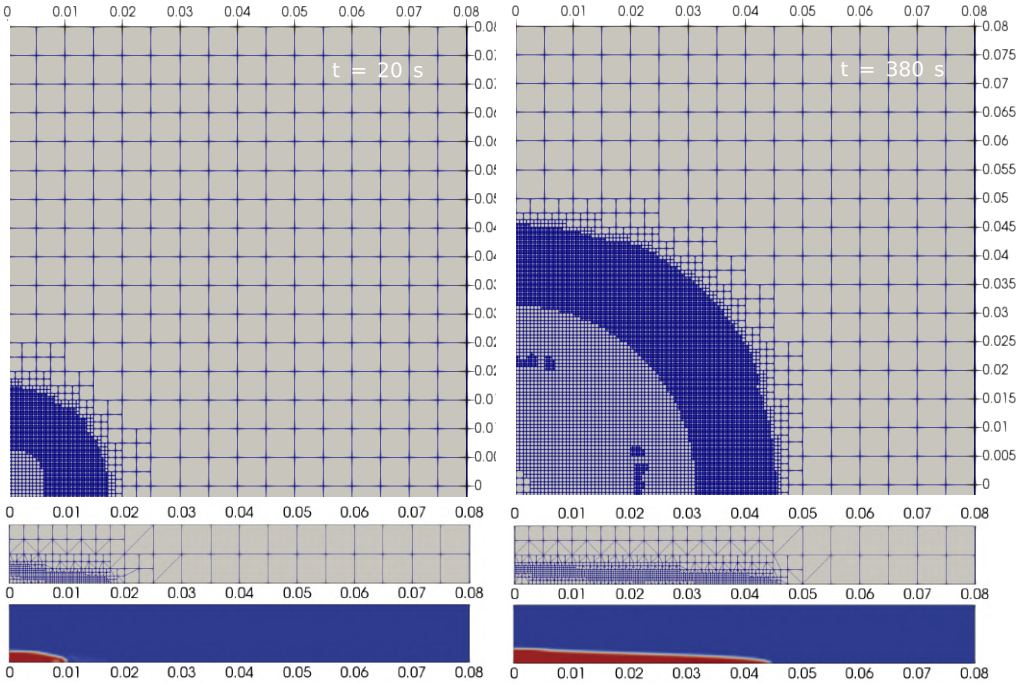


Figure 4.20: *Axisymmetric cooling and spreading.* Simulation computed with the dynamic mesh refinement up to 4 refinement levels over an initial uniform mesh defined by $\Delta x = 0.005\,\mathrm{m}$ represented at time $t = 20,\ 380\,\mathrm{s}$. *Top row:* bottom view of the discretization grid. *Middle row:* side view of the discretization grid on the plane $x = 0$. *Bottom row:* side view of the α-field on the plane $x = 0$.

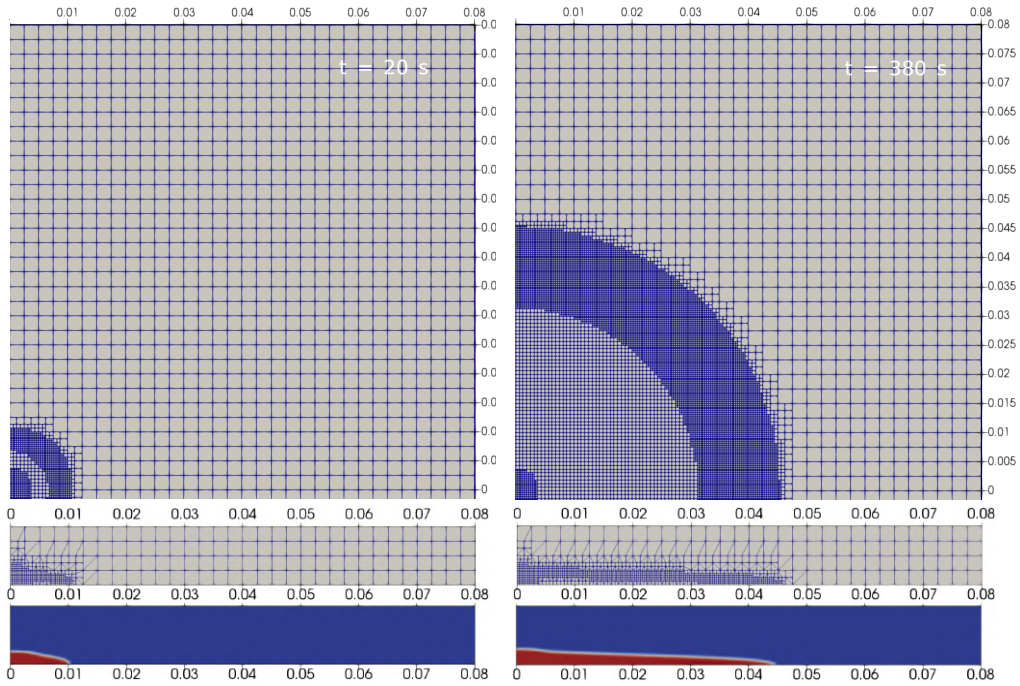


Figure 4.21: *Axisymmetric cooling and spreading.* Simulation computed with the dynamic mesh refinement up to 3 refinement levels over an initial uniform mesh defined by $\Delta x = 0.0025\,\mathrm{m}$ represented at time $t = 20,\ 380\,\mathrm{s}$. *Top row:* bottom view of the discretization grid. *Middle row:* side view of the discretization grid on the plane $x = 0$. *Bottom row:* side view of the α-field on the plane $x = 0$.

similar, since the simulation over the uniform static mesh required 5994.02 s and the other 8112.01 s. This is due to the fact that the 3rd refinement level occupies a big part of the computational domain (see Figure 4.19) because the fluid is very thin and the refinement is still relatively coarse, therefore there is no time saving in the use of the dynamic refinement; on the contrary, the opposite happens (and probably the computations for the adaptive refinement are relatively time-consuming). However, this phenomenon should be analyzed more deeply by further investigation and more simulations.

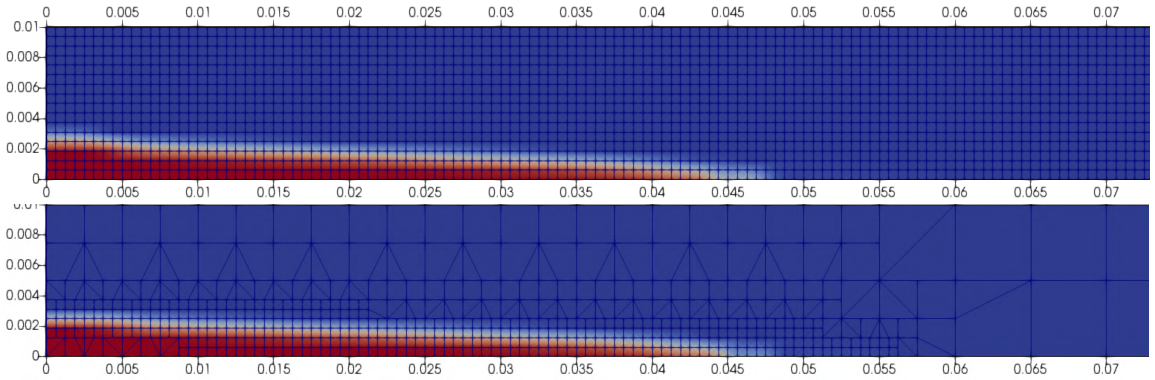


Figure 4.22: *Axisymmetric cooling and spreading.* Comparison between simulations computed with static (*top*) and dynamic (*bottom*) mesh. Side view of the α-field at time $t = 380\,\mathrm{s}$ on the vertical plane $x = 0$.

The second part of this section focuses on the parallel computation and the scalability performances of this test. For our parallel calculations, we used the High Performance Computing Super Micro cluster *laki* (16 nodes and 256 core Intel Xeon 2.40GHz, interconnection infiniband 40 Gbps at low latency) hosted at INGV, Section of Pisa. We parallelized the test that uses the adaptive mesh up to 4 refinement levels starting from an initial uniform grid with $\Delta x = 5 \cdot 10^{-3}\,\mathrm{m}$ (Figure 4.20). OpenFOAM parallelization relies on the domain decomposition and on the subsequent distribution of the fields. For our tests, we divided the domain (along the horizontal directions) into 2, 4, 8, and 16 pieces respectively. Table 4.4 reports the elapsed time needed to compute the simulations for each decomposition, and also the execution time necessary for the serial computation. As one would expect, the serial computation required the biggest amount of time with respect to the other cases. Moreover, the execution distributed on 4 cores used half of the time that took the computation to run over 2 cores. However, the reduction in the execution time does not continue increasing the cores, and the use of 8 and 16 cores requires more time than from the use of 4, even if this time is still less than that necessary for the serial computation. This limit in the scaling process is probably due to two factors. On one side, by increasing the number of cores, the number of cells per core decreases, whereas the communication between them increases. Other studies showed that with OpenFOAM the optimum cell count per core ranges from approximately 10,000 to approximately 50,000, and is highly dependent on the solver being used (the reader can refer to the study in Keough [146]). On the other side, the dynamics of this test is not distributed over all the domain, in particular in the first phases, and thus the computational load, with our choice for the domain decomposition, could not be optimal. OpenFOAM provides other options for the domain decomposition, that could possibly improve the parallel performance for this test.

Differently for the tests of the previous section, here we also modeled the cooling of

cores	decomposition	time
1	–	31486.72 s
2	2×1	28184.20 s
4	2×2	14115.90 s
8	4×2	26704.01 s
16	4×4	26682.20 s

Table 4.4: *Axisymmetric cooling and spreading.* Elapsed time for the sequential and parallel executions of simulations of 380 s.

the fluid while it spreads. For this reason, we conclude this section with an analysis on the temperature evolution of the flow, discussing the impact of the thermal heat loss processes on the final distribution of temperature inside the liquid. We considered once again the simulation obtained with adaptive mesh and that uses up to 4 refinement levels (results are shown in Figure 4.20), and we observed how the temperature distribution inside the liquid at time $t = 380$ s varies when considering all the heat loss processes (radiation and convection with the environment and conduction with the soil) or not. Figure 4.23 shows the fluid temperature with side views at $x = 0$ in the following circumstances, from top to bottom: (*i*) all the energy heat loss processes are active; (*ii*) radiation is suppressed, convection and conduction are active; (*iii*) radiation and convection are suppressed, only conduction is active; (*iv*) zoom at the front of the liquid in order to better appreciate the effects of conductive heat loss. We observe that conduction has the minor impact of the thermal heat loss with respect to the other processes. We remark that the viscosity is not temperature dependent, therefore the dynamics are not affected by different temperature distributions.

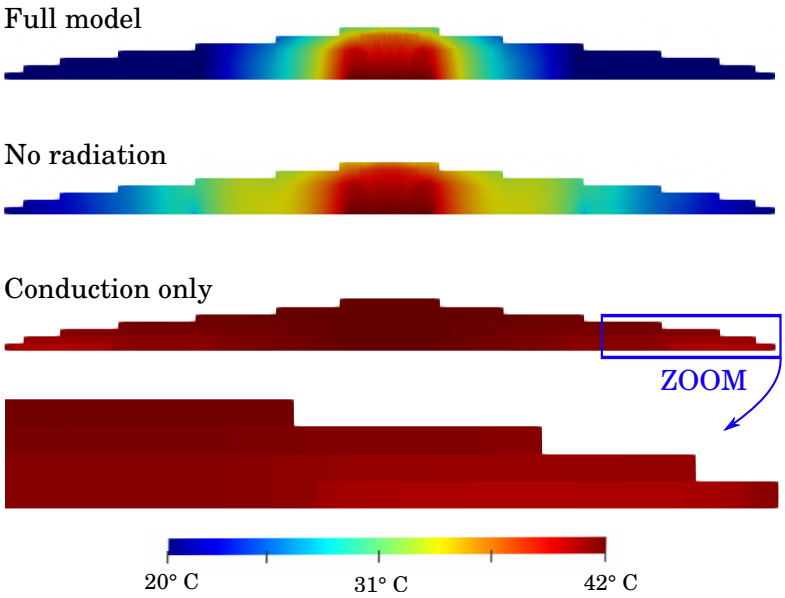


Figure 4.23: *Axisymmetric cooling and spreading.* Temperature distribution inside the liquid at time $t = 380$ s, side view at $x = 0$. A vertical larger scale was used to better see the vertical distribution.

Chapter 5

Conclusions, comparisons and future developments

The work presented in this thesis focused on developing mathematical and numerical models for viscous fluids, with particular attention on the application to the simulation of lava flows. With this aim in mind, we assumed the fluid of interest to be a hot and viscous fluid in a laminar regime that cools because of radiative and convective heat loss from the surface through the environment and because of conduction with the soil. Then, following two approaches, i.e., the depth-averaged (or shallow-water) approach and the 3D multiphase description, we obtained different systems of partial differential equations, requiring different numerical techniques, both based on the finite-volume method, to be solved. These numerical methods were exposed in detail in the thesis, together with the discussion of a comprehensive set of numerical simulations showing their effectiveness.

Unlike the classical shallow water equations, where a constant velocity profile is assumed, we developed a model that accounts for the possibility of having the velocity varying with the flow depth. In particular, we assumed a parabolic profile for the velocity to model a laminar viscous fluid over a topography, where a no-slip condition is prescribed. The model has also been enriched with an additional transport equation for a quantity whose values vary with the vertical flow coordinate. That quantity can be particle concentration in sediment-laden flows, where a high-concentration basal layer affects the transport dynamics or the temperature when a thermal boundary layer develops close to the bottom because of conduction. For this latest case, we assumed a piecewise-linear profile, and we added in the transport equations the appropriate terms describing radiative, convective, and conductive heat exchanges, together with viscous heating. From the assumptions of non-constant vertical profiles for velocity and temperature, we have shown that two coefficients appear in the hyperbolic term of velocity and temperature equations called shape factors (or Boussinesq factors). An in-depth study of the resulting partial differential equations was carried out through the characteristic analysis of the system, namely by computing the eigenvalues of the Jacobian matrices of the system (corresponding to the characteristic speeds of the flow) and showing the relationship between them and the shape factors.

The spatial discretization of the system of equations was achieved by using a modified version of the central-upwind Finite Volume Method scheme introduced by Kurganov and Petrova [158], originally developed to solve the classical shallow-water equations. We based the development of our numerical scheme on this method because of its relative simplicity of the implementation, the fact that, being a high-order scheme, presents a low

numerical diffusion, the positivity-preserving property, and its well-balancing property, which guarantees that the numerical scheme correctly describes steady-state solutions. Here, we have proven that for our modified system of equations and our modified numerical scheme, the well-balancing property and the positivity-preserving property are still guaranteed, but with an additional condition on the shape factors.

The temporal discretization has been obtained by an Implicit-Explicit Diagonally Implicit Runge-Kutta scheme that treats the hyperbolic terms explicitly and implicitly the viscous term in the momentum equation and the heat transfer terms of the temperature equation. We checked that this scheme accurately describes the wave propagation and treats well discontinuities, both of the solution and the topography. We have shown that a geometrical limiter adoption produces accurate results for all the tests performed and that high-viscosity fluids are more sensitive to the limiter chosen in terms of diffusive behavior. When considering constant or parabolic velocity profiles, the main differences have been observed for supercritical regimes (mainly associated with simulations of low-viscosity fluids); in particular, over long simulated times, the fluid propagates faster, assuming a constant velocity profile than assuming a parabolic profile. Similar differences have been found for the simulation of a fluid with a temperature-dependent viscosity so that the combined effects of temperature and velocity profiles are evident. Also, we have shown that using the set of variables $P = [w, U, V, T]$ to compute the interface reconstructions (related to our formulation of the spatial scheme) produces results very similar to those obtained using the variables $[w, hU, hV]$ (as in the original formulation of the scheme). In addition, there is the advantage of no need to adopt desingularization techniques when computing the flow velocities from the momentum components. We have also checked the proper description of the thermal processes by comparing our simulations with a laboratory experiment. Finally, we tested the impact of the rheological parameters of a Bingham temperature-dependent viscosity on the simulations of a realistic lava flow over the Fogo volcano topography. This last application of the model showed promising results, both in terms of real event description and of computational time required for the simulation (approximately 40 minutes for a run of 1 day with a grid with a horizontal resolution of 20 meters).

As regards the 3D modeling approach, a multiphase approach for immiscible and incompressible viscous fluids has been adopted. Even in this case, we based our model on an existing solver to which we added the relevant features (as a transport equation for the temperature with heat loss processes), adapting it to the flows of interests. This new model has been implemented within the well-established CFD tool OpenFOAM. Thanks to the open-source nature of the framework, this choice allowed us to take advantage of the solvers and libraries already available for multiphase flows and modify them to add the desired features. In addition, OpenFOAM fully supports parallel computing, and thus any new solver can take advantage of this capability. Within OpenFOAM, the spatial discretization of the governing equations system, still relying on the Finite Volume Method schemes, follows the segregated strategy for which the equations are solved sequentially. Because of the assumption of incompressibility, the conservation mass equation reduces to a kinematic constrain, so an additional equation for pressure is derived. Pressure is treated implicitly, and iterations are adopted to enforce the coupling between pressure and velocity fields.

The 3D approach requires solving the conservation equations in the whole domain, and thus both in the regions occupied by the viscous fluid and the surrounding air. For this reason, it was essential to choose an approach that could guarantee an accurate description

of the interface between the two fluids. The Volume of Fluid technique, based on the Interphase Capturing strategy, was used in this thesis to model the multiphase dynamics: an equation for the transport of the volumetric phase fraction is introduced, and the two fluids are treated as a single fluid whose properties (namely, density and viscosity) vary in space according to the volumetric fraction of each phase. The Multidimensional Universal Limiter for Explicit Solution (MULES) method was then used to help in keeping the phase interface sharper.

As previously stated, an energy equation has been implemented in the 3D model to describe both the transport and diffusion of thermal energy throughout the fluids, together with the radiative and convective heat loss at the interface between the phases. In addition, the heat conduction with the ground was instead modeled as a boundary condition. Finally, a library for the dynamic mesh refinement has been added to the solver, allowing to start a simulation with a relatively coarse mesh and refine it dynamically during the simulation in regions where smaller cells are needed (for example, close to the interface between the two phases).

For this model, some tests have been performed to look at the capability of the numerical model to keep a sharp interface, at the efficiency of the dynamic mesh refinement adapted to refine the cells straddling the phase interface and to analyze the performances of parallelization and scalability. The preliminary tests did so far highlight that, with a mesh appropriately fine, the numerical scheme manages to maintain an accurate description of the phase interface. Furthermore, we saw that by using the dynamic refinement, we obtained results that are perfectly comparable to those obtained with a uniform static mesh defined by the finest size of spatial discretization of the dynamic mesh, but with the critical difference of a sensible reduction in the computational time for the dynamic mesh case. However, even though this advantage on the execution time is true in most cases, we saw this might not occur for simulations where the cells involved by the refinement occupy a significant part of the domain.

By comparing the two physical and numerical models presented in this thesis, some differences may be highlighted and discussed. First of all, as it is evident, the use of a depth-averaged model does not allow a complete description of the vertical distribution of the flow variables, but it is based on assumptions for the vertical profiles. For example, we used a parabolic profile for the horizontal velocity in our model, based on the assumption of a well-developed laminar flow. However, more complex flow conditions and transient regimes (not resulting in a well-developed profile) could result in different velocity profiles. In these cases, a 3D approach is better suited to simulate the process of interest dynamics properly. The same is true for the temperature profile, which could differ substantially from the piecewise-linear profile assumed for our depth-averaged model. In addition, a 3D model can also better simulate the vertical variations in flow rheology, which can be important when the viscosity depends on temperature, for example. Generally, in depth-averaged models, the vertical distribution is not tied to the internal viscous forces but, as stated above, to pre-established model assumptions. We observe here that, in principle, it is possible to improve the description of the vertical profiles also used with depth-averaged models. Our next aim is to consider for them a dynamic evolution, allowing them to vary as a function of the local flow regime. For example, different velocity profiles can be designed as a function of the local Reynolds number, distinguishing between laminar and turbulent regions of the flow. The thermal boundary layer thickness discussed above could be related to the velocity boundary layer thickness, with the ratio of the two thicknesses governed by the Prandtl number (the ratio of momentum to thermal diffusivity). A similar

technique can also be used when modeling by a depth-averaged approach a sediment-laden, by defining the particle concentration profile as a function of the Rouse number [235, 262]. However, these modifications require not only a rewriting of the set of equations, since the depth-averaging of non-constant profiles gives rise to new shape factors in all the terms of the governing equations, but also further studies on the properties of the numerical schemes (well-balancing and positivity-preserving), which are not obvious when we modify the model.

Another difference between the models concerns the other hypotheses of the shallow water model that consist of assuming a negligible vertical velocity and hydrostatic pressure distribution. Under those conditions, the model cannot entirely capture fast dynamics that develop over terrain with high gradients. Natural processes that present those features are, for example, avalanches, landslides, and debris flow, but even lava flows may develop over steep terrain and flow fast. Hergarten [123] proposed a modified shallow water model that introduces a correction factor for the friction term in order to better capture dynamics over a steep slope. Xia and Liang [276] proposed another modified model that takes into account both the effects of the vertical acceleration and the effects of curvature due to a complex morphology of the terrain, introducing inside the momentum equation correction factors for the pressure and friction terms. As a future development, we plan to modify our model by introducing correction factors of this kind to make our model more flexible and applicable to a broader spectrum of cases.

Despite all the differences presented so far, depth-averaged models have proven to be accurate enough for many geophysical applications (landslides, tsunamis, lava flows, lahars) to keep being a still convenient simplification in terms of computational cost with respect to 3D models. In fact, at present, the computational time required for 3D simulation remains prohibitive for real-time applications and for the use of numerical simulations in rapid hazard assessment and quantification. From the computational point of view, one of the advantages of the depth-averaged approach is that the friction/viscous forces are not modeled through a second-order differential term but through a simpler (and non-differential) term. The implicit treatment of the viscous forces, in the 3D case, requires instead the solution of a large coupled system of equations resulting from the implicit spatial discretization of the second-order differential term, whereas, in the depth-averaged case, there is no coupling between the equations associated to these terms. That allowed us to adopt the IMEX scheme, where the implicit viscous terms are integrated cell by cell, significantly reducing the complexity of the problem. On the other hand, the OpenFOAM approach to reduce the complexity of the solution of the implicit terms is based on a segregated procedure, where the different governing equations are solved sequentially in an iterative way. We also observe that both the models are based on a finite-volume discretization of the spatial differential terms, which is an optimal choice for the solution of equations based on conservation laws, because it enforces conservation of quantities also at the discretized level.

Finally, as previously stated, we notice that the 3D model we developed can employ both the dynamic mesh refining and the parallelization capabilities already available in OpenFOAM, which at present are not implemented in the depth-averaged model that has been developed in a different framework. We remark that this is not an intrinsic advantage of the 3D approach with respect to the depth-averaged approach but comes from the particular choices we made so far on the numerical codes for our models.

In addition to the differences associated with the different modeling approaches, some model-specific limits are associated with the particular implementation of the numerical

model.

For the depth-averaged model, the numerical scheme is based on a time step controlled by the CFL condition, where the relevant velocity is the maximum (in magnitude) eigenvalue of the flux Jacobian, i.e., the maximum characteristic velocity. For high-viscosity flows (for example, lava flows), this can be a lot larger than the flow velocity, resulting thus in time steps much smaller than those obtained with numerical schemes for which the CFL condition is applied to the flow velocity. The PISO scheme adopted by the 3D model represents an example of this latter approach, allowing the adoption of larger time steps than the depth-averaged model. In addition, whereas the choice of the Kurganov-Petrova scheme is well suited for the discretization of conservative terms, the generalization to non-conservative differential terms could arise for more complex models (for example, when considering multi-layer depth-averaged models) is not obvious.

For the 3D numerical model we developed, even though the preliminary results obtained are acceptable, some modifications could be made to increase its flexibility and capability to better capture the effects of thermal processes on the dynamics. In this regard, a first limit comes with the mesh: the user should pay attention to have grid cells with an aspect ratio as close as possible to 1 in order o have both a good description of the phase interface and proper modeling of the heat loss processes occurring at the interface between the phases. The radiative and convective terms computation requires the surface area of the phase interface, which translates into knowing the surface area of each interface cell. At present, we use approximation to compute the surface area from the volume, which is based on the assumption of an aspect ratio close to 1, as suggested by Almeland [2]. When the cell aspect ratio is smaller or larger than 1, the surface area measure could be overestimated or underestimated, with a consequent error in the computation of the heat loss. Further work needs to be done to better estimate the interface surface area with different strategies, independent from the cells aspect ratio, but based on the gradients of the volumetric phase fraction field, for example.

Lastly, we touch on a further problem related to the 3D solver that we did not mention before but has received considerable attention: the development of spurious velocities in the low-density fluid near the phase interface. In the first place, such phenomenon was related to the surface tension [190, 218]; instead, more recently, it was demonstrated that the development of spurious velocities takes place even in situations where the surface tension is suppressed [265, 271]. In particular, the work by Eltard Larsen et al. [78] attributed the growth of the spurious velocities to the high ratio liquid/gas density, whereby even small erroneous transfers of momentum across the interface from the heavier to the lighter fluid cause a considerable acceleration of the lighter fluid, as described and discussed also by Vukcevic [264]. Even though the spurious currents represent an undesirable and unphysical process, it does not affect the dynamics of the fluid of interest that is often the heavier one. Instead, in the case that an accurate description of the velocity field of the lighter fluid is of interest, the spurious currents represent a non-negligible problem.

We end this section by considering one of the initial objectives of this thesis: the development of depth-averaged and 3D models for the simulation of lava flows. For the depth-averaged model, we have shown an application to the simulation of the first day of the 2014-2015 Fogo eruption. We used a real topography and a Bingham rheological model with temperature-dependent viscosity for this simulation, obtaining a good agreement with the observations. Instead, for the 3D model, we have presented only preliminary results from standard benchmarks, and we need to deepen the study and test the model on more cases. Concerning the simulations presented here, the model already allows the

use of a temperature-dependent viscosity model and simulates a flow over topography, but a complete simulation of a lava flow accounting for these features, with a source, and with the dynamic mesh, still needs some work. Finally, remaining in the context of lava flows, we observe that the process of crystallization induced by fluid cooling has a high impact on flow dynamics because it leads to a viscosity increase and a possible flow stopping. This phenomenon can be treated by including the transport of an additional *dispersed phase*, that represents crystals, in our model, together with a source term modeling the crystallization rate as a function of flow temperature and magma composition. Our future plan is to include this process in our models, both in the depth-averaged and in the 3D, together with a rheological model capable of taking into account both temperature and crystal content.

To conclude, we remark that to evaluate our models' capabilities better to describe properly the physical processes treated, we would need to compare the results of our simulations with more analog laboratory experiments, for which the flow conditions are well constrained. In this context, a collaborative relationship with a research group that works on laboratory experiments or large-scale experiments for lava flows would be desirable. A close collaboration could result in a better identification of the parameters to measure during the experiments that could be compared with the outputs of the numerical simulations. In addition, this collaboration would help better understand the main processes governing the dynamics of the flows of interest, and thus on the terms to be introduced in the models.

Appendix A

OpenFOAM tutorial: add energy equation with radiative, convective and conductive heat loss to interFoam

This tutorial walks the reader through all the steps to create a new solver based on `interFoam` (OpenFOAM-v1912) by adding thermal properties and an energy equation to model a hot fluid that exchanges energy with the environment. The theoretical bases behind the solver are described in §2.2 as far as the physical model concerns, in Chapter §4 for the numerical contents, whereas the basics of the computational setting of OpenFOAM are described in §1.3. Our work is founded on a tutorial for OpenFOAM 4.1 by Hannah Dietterich taken at IAVCEI 2017, OpenFOAM Workshop, and on the work of Almeland [2].

`interFoam` is a multiphase, transient solver for two immiscible, incompressible, isothermal, viscous fluids and performs both in laminar and turbulent regime. The different fluid properties are arranged by the libraries for the transport models, i.e. `incompressibleTwoPhaseMixture` and `immiscibleIncompressibleTwoPhaseMixture`. The incompressible Navier-Stokes equations are solved together with an equation for the volumetric phase fraction. The momentum-pressure system of coupled equations is solved in a segregated fashion using the PIMPLE algorithm. The multiphase model adopts the Volume of Fluid method to capture the interface between the fluids. The numerical scheme MULES (Multidimensional Universal Limiter for Explicit Solution) is a Flux-Corrected Transport technique adopted to reduce the numerical diffusion of the interface between the phases.

Our work fits and starts into this context. The energy equation that we add presents radiative and convective heat loss terms referred at the free surface meanwhile the conductive heat exchange with the soil is implemented as a boundary condition. Instead of presenting in a single moment all the changes necessary to obtain the new solver, the process is divided into different, sometimes disjointed, steps. So, starting from `interFoam`, we will produce the following solvers:

- `interThermalFoam`;
- `interThermalRadFoam`;
- `interThermalRadConvFoam`.

This strategy should help the reader to understand better the description and motivation behind any modifications. Besides, this facilitates the reader that is interested in specific

steps to focus only on them. Furthermore, after the implementation of each solver, it is tested with the dam break (`damBreak`) simulation which is a classic test case for `interFoam`, already available to use. We modify the original test into `damBreakAddT` in order to test our solvers. We also highlight that in the rest of the work the symbol $ indicates commands in the terminal, while Courier font indicates text within files.

The Tutorial is structured as follows. OpenFOAM is devoid of a graphical interface, and both data and settings are saved as text files which are named and located in a very precise way based on a predefined structure. Hence, before starting with the Tutorial instructions, we spend some words in §A.1 to describe the structure of the main folders and files constituting the software. §A.2 has preliminary instructions about how to install OpenFOAM by the Docker container. In §A.3, a basic energy equation with the hyperbolic transient terms and the diffusion term is added to the solver and the transport models are modified in order to take into account also the necessary parameters bounded to the energy model. In §A.4 and §A.5 we introduce radiative and convective terms respectively into the energy equation; these sections could be independent, but they share some contents exposed only in §A.4. Finally, §A.6 describes how to implement the boundary conditions for the conduction between the fluid and the surface on which flows.

A.1 OpenFOAM structure

The main folders are organized as represented in Figure A.1. OpenFOAM is used primarily to create executables, known as applications, that fall in two categories: the solvers, that are designed to solve continuum mechanics problems, and the utilities, that perform data manipulation, meshing creation, case setup, solution monitoring, data export and data visualization. The fact that OpenFOAM is accompanied by pre- and post-processing environments ensures consistent data handling across all environments. The directory run/tutorials has the same structure as the folder applications/solver, but it contains the directories devoted to the simulations.

Each simulation have a directory, say it case, with some sub-directories, as those reported in Figure A.2. Originally the case directory contains only the sub-directories 0, constant and system. The folder 0 has got the initial and boundary conditions of the variables involved into the dynamics, whereas the other *time directories* are created at run time to store the fields computed by the solver at those time steps. The constant directory initially contains the definition of the gravity acceleration and the dictionaries for the kinematic and viscous transport properties. At run time, after the mesh creation, the folder contains also polyMesh, a folder in which all the elements of the mesh are defined and stored (points, cells, faces, ...). The system directory contains: the settings for the run in the file controlDict, the dictionary blockMeshDict to create the mesh, the settings for the discretization schemes fvSchemes and for the solution procedure fvSolution. At run time, the solver that is called to compute the solution reads these files in the case directory and runs the simulation according with those settings.

From a mathematical point of view, OpenFOAM solves a system of PDEs by using the Finite Volume Method on unstructured meshes and it reduces the problem to a linear system of equations that solves iteratively.

Even though OpenFOAM offers solvers for different systems of PDEs, the user may not find e perfect match with the problem he aims to solve, this was also our situation. However, the open-source nature of the code allows the user to overcome this difficulty permitting to build a new solver for the application one wish, and meanwhile, take advan-

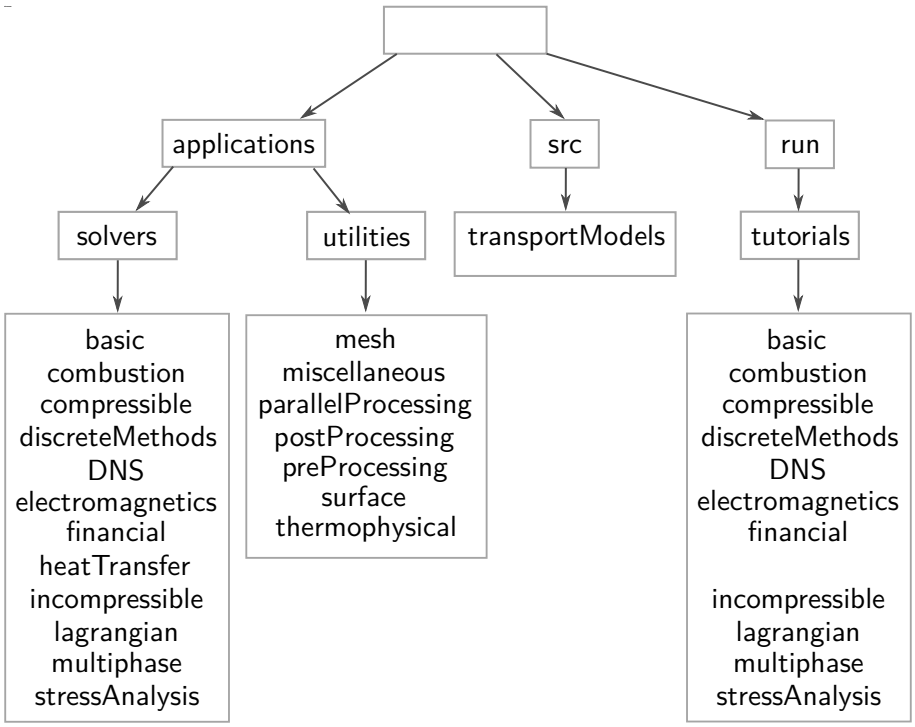


Figure A.1: Structure of the main folders of OpenFOAM.

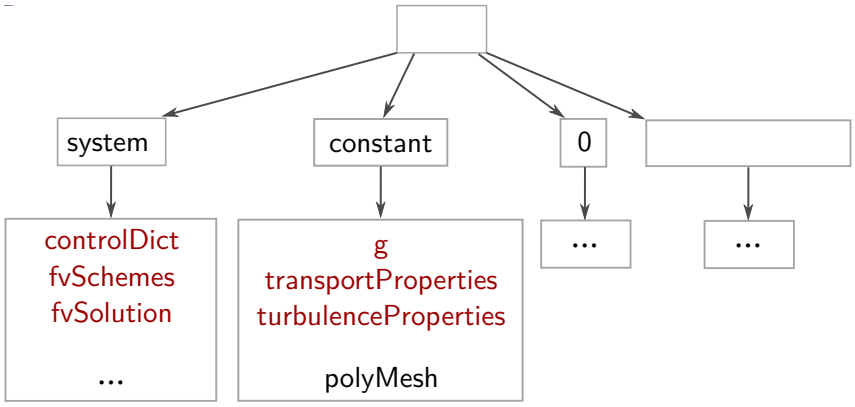


Figure A.2: Structure of the case folder.

tage both of the huge numbers of schemes and methods already implemented and ready to use and also of the good properties of the software. The user can set up easily almost every kind of computational domain of arbitrary geometry, from the simplest uniform grids to the most complicated unstructured polyhedral meshes. The basic mesh generator is blockMesh, whereas the utility snappyHexMesh allows modifying an already existing mesh into a more sophisticated one guaranteeing a minimum mesh quality. The dictionaries that the user can set to create the mesh are blockMeshDict and snappyHexMeshDict respectively, both located in the case/system folder.

A.2 Getting started with OpenFOAM

OpenFOAM-v1912 can be installed with Docker on Linux by following the instructions at the link openfoam.com/download/install-binary-linux.php. After that, a new folder

named OpenFOAM should be present in your home.

(1) Start OpenFOAM in your terminal window, by the Docker container, with the script:

```
$ cd ~/OpenFOAM/installation1912
$ ./startOpenFOAM
```

This will open a new shell with the OpenFOAM environment fully installed and ready to use.

(2) Create a directory for the user modifications (if not already present) which reflects the same structure and directory subdivisions as in OpenFOAM:

```
$ mkdir OpenFOAM/username-v1912
$ cd OpenFOAM/username-v1912
$ mkdir applications/solvers src run/tutorials
```

The user has some shortcuts, written in capital letters, to access some main folders from the terminal, as we will see.

A.3 Add Energy Equation

A.3.1 Modifying the transport model to include thermal parameters

The so called `transportModel` is a base-class for all the transport models used by the incompressible and turbulence models. The classes derived from this one can deal with both multiphase or single phase conditions. The `transportModel` class owns some public member functions, defined as `virtual` and therefore implemented by its derived classes, that are: `nu()`, returning the laminar viscosity, `correct()`, that makes a correction on the laminar viscosity, and `read()` that reads the transportProperties dictionary located inside the folder of the test case case/constant.

In order to add the thermal parameters to our solver, we modify only two classes: `incompressibleTwoPhaseMixture`, which is derived from the `transportModel` class, and `immiscibleIncompressibleTwoPhaseMixture` class, that is derived from the previous one.

(3) Copy the transportModels directory into your personal src folder.

```
$ cp -r $FOAM_SRC/transportModels/incompressible src/.
$ cp -r $FOAM_SRC/transportModels/
   immiscibleIncompressibleTwoPhaseMixture src/.
```

(4) Inside the directory incompressible, modify the files related to the class incompressibleTwoPhaseMixture starting from editing the names of the header.

```
$ cd $WM_PROJECT_USER_DIR/src/transportModels/incompressible
$ mv incompressibleTwoPhaseMixture addTIncompressibleTwoPhaseMixture
$ cd addTIncompressibleTwoPhaseMixture
$ mv incompressibleTwoPhaseMixture.H addTIncompressibleTwoPhaseMixture.H
$ mv incompressibleTwoPhaseMixture.C addTIncompressibleTwoPhaseMixture.C
$ sed -i s/incompressibleTwoPhaseMixture/
   addTIncompressibleTwoPhaseMixture/g addTIncompressibleTwoPhaseMixture
   .*
```

Edit Make/files: open it and update the names of files and directories, and indicate the library created by the user instead of the default one

```
// End add
LIB = $(FOAM_USER_LIBBIN)/libaddTIncompressibleTransportModels
```

Edit Make/options: open it and change the link to twoPhaseMixture:

```
-I$(LIB_SRC)/transportModels/twoPhaseMixture/lnInclude
```

(5) Edit immiscibleIncompressibleTwoPhaseMixture class (pay attention to 'i' and 'I'): change the name to the folder, to the files, and to all the occurrences.

```
$ cd $WM_PROJECT_USER_DIR/src/transportModels
$ mv immiscibleIncompressibleTwoPhaseMixture
   addTImmiscibleIncompressibleTwoPhaseMixture
$ cd addTIncompressibleTwoPhaseMixture
$ mv immiscibleIncompressibleTwoPhaseMixture.H
   addTImmiscibleIncompressibleTwoPhaseMixture.H
$ mv immiscibleIncompressibleTwoPhaseMixture.C
   addTImmiscibleIncompressibleTwoPhaseMixture.C
$ sed -i s/immiscibleIncompressibleTwoPhaseMixture/
   addTImmiscibleIncompressibleTwoPhaseMixture/g
   addTImmiscibleIncompressibleTwoPhaseMixture.*
$ sed -i s/incompressibleTwoPhaseMixture/
   addTIncompressibleTwoPhaseMixture/g
   addTImmiscibleIncompressibleTwoPhaseMixture.*
```

Edit Make/files: open it, update the names of files and directories, and indicate the library created by the user instead of the default one.

```
addTImmiscibleIncompressibleTwoPhaseMixture.C

LIB = $(FOAM_USER_LIBBIN)/libaddTImmiscibleIncompressibleTwoPhaseMixture
```

Edit Make/options: change the relative link to incompressible folder in EXE_INC.

```
-I$(WM_PROJECT_USER_DIR)/src/transportModels/incompressible/lnInclude \

-I$(WM_PROJECT_DIR)/src/transportModels/interfaceProperties/lnInclude \
-I$(WM_PROJECT_DIR)/src/transportModels/twoPhaseMixture/lnInclude \
```

Finally, add to LIB_LIBS the next lines.

```
-L$(FOAM_USER_LIBBIN) \
-laddTIncompressibleTransportModels \
```

(6) Inside the header file addTIncompressibleTwoPhaseMixture.H, add the declarations of new variables. We introduce the heat capacity cp and the Prandtl number Pr for both phases and then the face-interpolated conductivity field kappaf.

```
dimensionedScalar rho1_;
dimensionedScalar rho2_;
// Add
dimensionedScalar cp1_;
dimensionedScalar cp2_;
dimensionedScalar Pr1_;
dimensionedScalar Pr2_;
// End Add
```

AND

```
const dimensionedScalar& rho2() const
    {
        return rho2_;
    };

// Add
// Return const-access to phase1 heat capacity
const dimensionedScalar& cp1() const
    {
        return cp1_;
    }
// Return const-access to phase2 heat capacity
const dimensionedScalar& cp2() const
  {
      return cp2_;
  };
// Return const-access to phase1 Prandtl number
const dimensionedScalar& Pr1() const
  {
      return Pr1_;
  }
// Return const-access to phase2 Prandtl number
const dimensionedScalar& Pr2() const
  {
      return Pr2_;
  };
// End Add
```

AND

```
tmp<surfaceScalarField> nuf() const;

// Add
// Return the face-interpolated conductivity
tmp<surfaceScalarField> kappaf() const;
// End Add
```

(7) Add the initialization of the thermal parameters and the implementation of the new member functions to the source file addTIncompressibleTwoPhaseMixture.C.

```
rho2_("rho", dimDensity, nuModel2_->viscosityProperties()),

// Add
cp1_("cp", dimensionSet(0, 2, -2, -1, 0, 0, 0), nuModel1_->
   viscosityProperties()),
cp2_("cp", dimensionSet(0, 2, -2, -1, 0, 0, 0), nuModel2_->
   viscosityProperties()),

Pr1_("Pr", dimensionSet(0, 0, 0, 0, 0, 0, 0), nuModel1_->
   viscosityProperties()),
Pr2_("Pr", dimensionSet(0, 0, 0, 0, 0, 0, 0), nuModel2_->
   viscosityProperties()),
// EndAdd
```

AND

```
Foam::tmp<Foam::surfaceScalarField>
Foam::incompressibleTwoPhaseMixture::nuf() const
{
```

```
    ...
}

// Add
Foam::tmp<Foam::surfaceScalarField>
Foam::incompressibleTwoPhaseMixture::kappaf() const
{
    const surfaceScalarField alpha1f
    (
        min(max(fvc::interpolate(alpha1_), scalar(0)), scalar(1))
    );

    return tmp<surfaceScalarField>
    (
        new surfaceScalarField
        (
            "kappaf",
            (
                alpha1f*rho1_*cp1_*(1/Pr1_)
    *fvc::interpolate(nuModel1_->nu())
                + (scalar(1) - alpha1f)*rho2_*cp2_
    *(1/Pr2_)*fvc::interpolate(nuModel2_->nu())
            )
        )
    );
}
// End Add
```

AND

```
nuModel1_->viscosityProperties().readEntry("rho", rho1_);
nuModel2_->viscosityProperties().readEntry("rho", rho2_);

// Add
nuModel1_->viscosityProperties().readEntry("cp", cp1_);
nuModel2_->viscosityProperties().readEntry("cp", cp2_);

nuModel1_->viscosityProperties().readEntry("Pr", Pr1_);
nuModel2_->viscosityProperties().readEntry("Pr", Pr2_);
// End Add
```

(8) With the previous steps, the libraries of both classes have been updated, so now it is necessary to compile them. Navigate to the incompressible directory and then compile creating the library for addTIncompressibleTransportModels.

```
$ cd $WM_PROJECT_USER_DIR/src/transportModels/incompressible
$ wclean
$ wmake libso
```

After, navigate back to the addTImmiscibleIncompressibleTwoPhaseMixture directory and compile also that library.

```
$ cd ../addTImmiscibleIncompressibleTwoPhaseMixture
$ wclean
$ wmake libso
```

The new libraries are located in $FOAM_USER_LIBBIN, while new folders lnInclude has been created in each directory in which the compiling processes have been executed.

A.3.2 Modifying the `interFoam` solver

At point (2), the path for the directory solvers was created, and we continue to establish the correct path to append the new solver `interThermalFoam`. Furthermore, also the folder VoF, concerning the implementation of the Volume of Fluid (VoF) method, must be copied there and then modified.

```
$ cd $WM_PROJECT_USER_DIR/applications/solvers
$ mkdir multiphase
$ cd multiphase
$ cp -r $FOAM_SOLVERS/multiphase/VoF .
$ mkdir interThermalFoam
$ cd interThermalFoam
$ cp -r $FOAM_SOLVERS/multiphase/interFoam/ .
```

We remove the extraneous solvers, namely those we do not use, that are present inside the interFoam folder, and then we modify the solver itself starting from its name and all the occurrences of its name inside the code.

```
$ rm -r overInterDyMFoam
$ rm -r interMixingFoam
$ mv interFoam.C interThermalFoam.C
$ sed -i s/interFoam/interThermalFoam/g interThermalFoam.C
$ sed -i s/immiscibleIncompressibleTwoPhaseMixture/
   addTImmiscibleIncompressibleTwoPhaseMixture/g interThermalFoam.C
```

(10) Edit the file Make/files: open and update it by renovating the solver name and the position in which locate the executable.

```
interThermalFoam.C
EXE = $(FOAM_USER_APPBIN)/interThermalFoam
```

(11) Edit the file Make/options: open and update it. First, update in `EXE_INC`:

```
-I$(WM_PROJECT_USER_DIR)/applications/solvers/multiphase/VoF \
-I$(WM_PROJECT_USER_DIR)/src/transportModels/incompressible/lnInclude \
-I$(WM_PROJECT_USER_DIR)/src/transportModels/
   addTImmiscibleIncompressibleTwoPh
aseMixture/lnInclude \
```

Second, modify `EXE_LIBS` by adding and updating the file with the following lines:

```
-L$(FOAM_USER_LIBBIN) \
-laddTIncompressibleTransportModels \
-laddTImmiscibleIncompressibleTwoPhaseMixture \
```

(12) Inside the folder interThermalFoam, modify the module createFields.H file by adding a scalar field for temperature and the others thermal variables. The temperature field added is called `T`, and is related to the cell centroids since it is defined as a volume scalar field. Also, it is an input/output object that must be read at the initial time and that must be written at every output time. The specific heat of each phase is denoted as `cp1` and `cp2`, and are both read from the transport properties dictionary; these values are important for the creation of other two fields `rhoCp` and `rhoCpPhi` used for the discretization of the transient term and of the advective term, respectively, that are located inside the energy equation.

```
Info<< "Reading field U\n" << endl;
volVectorField U
(  ...  );

// Add
Info<< "Reading field T\n" << endl;
volScalarField T
(
    IOobject
    (
        "T",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);
// End Add
```

AND

```
const dimensionedScalar& rho1 = mixture.rho1();
const dimensionedScalar& rho2 = mixture.rho2();
// Add
const dimensionedScalar& cp1 = mixture.cp1();
const dimensionedScalar& cp2 = mixture.cp2();
// End Add
```

AND

```
// Need to store rho for ddt(rho, U)
volScalarField rho
(
    IOobject
    (
        "rho",
        runTime.timeName(),
        mesh,
        IOobject::READ_IF_PRESENT
    ),
    alpha1*rho1 + alpha2*rho2
);
rho.oldTime();

// Add
// Need to store rhoCp for ddt(rhoCp, T)
volScalarField rhoCp
(
    IOobject
    (
        "rhoCp",
        runTime.timeName(),
        mesh,
        IOobject::READ_IF_PRESENT
    ),
    alpha1*rho1*cp1 + alpha2*rho2*cp2
);
rhoCp.oldTime();
// End Add
```

AND

```
// Mass flux
surfaceScalarField rhoPhi
(
    IOobject
    (
        "rhoPhi",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    fvc::interpolate(rho)*phi
);

// Add
// Need to store rhoCpPhi for div(rhoCpPhi,T)
surfaceScalarField rhoCpPhi
(
    IOobject
    (
        "rhoCpPhi",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    fvc::interpolate(rhoCp)*phi
);
// End Add
```

AND

```
// MULES compressed flux is registered in case scalarTransport FO needs
   it.
surfaceScalarField alphaPhiUn
(  ...  );

// Add
// global for TEqn use
tmp<surfaceScalarField> tPhiAlpha;
// End Add
```

(13) Move into the folder VoF and modify the files concerning the Volume of Fluid method.

```
$ cd ../VoF
```

We need to edit alphaEqn.H which contains the implementation of the numerical scheme that solves the α-equation. Add these lines to alphaEqn.H.

```
    if
    (
        word(mesh.ddtScheme("ddt(rho,U)"))
     == fv::EulerDdtScheme<vector>::typeName
     || word(mesh.ddtScheme("ddt(rho,U)"))
     == fv::localEulerDdtScheme<vector>::typeName
    )
    {
```

```
        rhoPhi = alphaPhi10*(rho1f - rho2f) + phiCN*rho2f;
        // Add
        rhoCpPhi = alphaPhi10*(rho1f*cp1 - rho2f*cp2) + phiCN*rho2f*cp2;
        // End Add
    }
```

AND

```
    else
    {
        if (ocCoeff > 0)
        {
            // Calculate the end-of-time-step alpha flux
            alphaPhi10 =
                (alphaPhi10 - (1.0 - cnCoeff)*alphaPhi10.oldTime())/
   cnCoeff;
        }

        // Calculate the end-of-time-step mass flux
        rhoPhi = alphaPhi10*(rho1f - rho2f) + phi*rho2f;
        // Add
        rhoCpPhi = alphaPhi10*(rho1f*cp1 - rho2f*cp2) + phi*rho2f*cp2;
        // End Add
    }
```

(14) Modify also the file alphaEqnSubCycle.H by adding the following line:

```
rho == alpha1*rho1 + alpha2*rho2;
// Add
rhoCp = alpha1*rho1*cp1 + alpha2*rho2*cp2;
// End Add
```

(15) Go back to the folder of our solver interThermalFoam and create a new file for the temperature equation called TEqn.H.

```
$ cd ../interThermalFoam
$ touch TEqn.H
```

Fill the file with the following lines.

```
surfaceScalarField kappaf = mixture.kappaf();

fvScalarMatrix TEqn
(
    fvm::ddt(rhoCp,T)
    + fvm::div(rhoCpPhi,T)
    - fvm::laplacian(kappaf,T)
);
TEqn.relax();
TEqn.solve();
```

(16) Update the solver itself, namely edit the file interThermalFoam.C, to include the new thermal parameters and to solve the energy equation. Add the call to the energy equation just before the call to the function `runTime.write()`

```
// Add
#include "TEqn.H"
// End Add
```

```
runTime.write();
```

(17) Compile it from inside the folder interThermalFoam.

```
$ wclean
$ wmake
```

A.3.3 Run a test case: dam break with temperature

The dam break case is a classical test for interFoam and we adopt its 2D frame for our preliminary tests.

(18) Copy the damBreak folder case from the original location into the directory run by creating the necessary path that reproduces the original position.

```
$ cd $FOAM_RUN/tutorials
$ mkdir multiphase/interFoam/laminar/damBreak/damBreakAddT
$ cd multiphase/interFoam/laminar/damBreak/damBreakAddT
$ cp -r $FOAM_TUTORIALS/multiphase/interFoam/laminar/damBreak/damBreak/
   .
```

(19) The folder 0 of the test case contains files that specify the initial and boundary conditions of the variables. Therefore, create the file with the conditions for the temperature, called T, by copying and modifying an already existing file, for example, the file with the conditions for velocity, named U.

```
$ cd 0
$ cp U T
```

Open the file T and do what stated in the following.

- Change the `class` name to `volScalarField`.
- Change the `object` name to `T`.
- Specify that temperature is expressed in Kelvin `[0 0 0 1 0 0 0]`.
- Set the `internalField` to `uniform 293`.
- Make all the boundary conditions `zeroGradient` except for the empty one.

(20) Update the transportProperties file, located in the constant folder, by adding values and units for the heat capacity and the Prandtl number of both phases.

```
$ cd ../constant
```

- water

```
cp              [0 2 -2 -1 0 0 0] 4190;
Pr              [0 0 0 0 0 0 0]   10.0;
```

- air

```
cp              [0 2 -2 -1 0 0 0] 1000;
Pr              [0 0 0 0 0 0 0]   0.72;
```

(21) Update the files inside the directory system.

```
$ cd ../system
```

In the controlDict, file update the application name.

```
application     interThermalFoam;
```

Inside the fvSchemes file, add a line in `divSchemes` that specifies the numerical schemes to use for the advective term of the energy equation:

```
divSchemes
{
    div(rhoPhi,U)  Gauss linearUpwind grad(U);
    div(phi,alpha)  Gauss vanLeer;
    div(phirb,alpha) Gauss linear;
    div(((rho*nuEff)*dev2(T(grad(U))))) Gauss linear;
    // Add
    div(rhoCpPhi,T) Gauss limitedLinear 1;
    // End Add
}
```

In fvSolution, copy the solver for `p_rgh` and modify it:

```
    // Add
    T
    {
        solver          PBiCG;
        preconditioner  DILU;
        tolerance       1e-07;
        relTol          0;
    }
    // End Add
```

In setfieldsDict, add the values for temperature:

```
defaultFieldValues
(
    volScalarFieldValue alpha.water 0
    volScalarFieldValue T 293 // Add
);

regions
(
    boxToCell
    {
        box (0 0 -1) (0.1461 0.292 1);
        fieldValues
        (
            volScalarFieldValue alpha.water 1
            volScalarFieldValue T 400  // Add
        );
    }
);
```

(22) Run the case from the main folder damBreakAddT. The executable Allrun launches three commands: first it creates the mesh with the utility blockMesh, then the utility setFields modifies the initialization of the fields `alpha` and `T`, and finally the solver `interThermalFoam` is executed.

```
$ cd ..
$ ./Allrun
```

(23) In the case folder, a number of new directories are created with files that store fields for every output time, according to what was initially set inside the dictionary system/controlDict. Use ParaView to see the results by launching this last command.

```
$ parafoam
```

A.4 Add the Radiative Heat Exchange term

With the previous modifications, we have added to the solver the equation for the transport and diffusion of the heat per unit volume (we report also the unit in the form expressed by OpenFOAM which consists of the sequence [kg m s K mol A cd])

$$\frac{\partial(\rho C_p T)}{\partial t} + \nabla \cdot (\rho C_p T \mathbf{u}) - \Delta(k_f T) = 0, \qquad [1, -1, -3, 0, 0, 0, 0],$$

where ρ kg m^{-3} is the density, C_p m^2 s^{-2} K^{-1} is the specific heat, k_f is the face-interpolated conductivity. The term of the heat radiation we are going to add depends on the fourth power of temperature and is proportional to several variables: the emissivity ϵ, the Stephan-Boltzmann constant $\sigma_{SB} = 5.67 \cdot 10^{-8}$ kg s^{-3}K^{-4}, the fractional area of the exposed inner core f, and the radiative free surface area A_{fs} m^2:

$$\epsilon \sigma_{SB} f A_{fs} \big(T^4 - T_{env}^4\big), \quad [1, 2, -3, 0, 0, 0, 0].$$

In order to add this kind of term to our equation, we have to divide it by the volume Vol m^3 because the equation is expressed per unit volume, then we get the next expression for the energy equation:

$$\frac{\partial(\rho C_p T)}{\partial t} + \nabla \cdot (\rho C_p T \mathbf{u}) - \Delta(k_f T) = -\frac{\epsilon \sigma_{SB} f A_{fs}}{Vol} \big(T^4 - T_{env}^4\big), \quad [1, -1, -3, 0, 0, 0, 0].$$

In order to include this source term into the implementation of the energy equation, we edit the file TEqn.H which represents this equation, but the definition and implementation of the source term are entrusted to the transport model addTImmiscibleIncompressibleTwoPhaseMixture that will be modified as well.

A.4.1 Editing the transport model

The radiative source term is naturally implemented by modifying the class `addTImmiscibleIncompressibleTwoPhaseMixture` since this class is derived both from the `addTIncompressibleTwoPhaseMixture` class and `interfaceProperties` class and this last one will be useful exactly because the radiative heat exchange is strictly confined at the interface between the two phases.

Local copy of the new transport model

Start by copying the directory of the transport model to edit and by changing the file name and of all its occurrences inside the file addTRadImmiscibleIncompressibleTwoPhaseMixture.

```
$ cd ~/OpenFOAM/username-v1912/src/transportModels
$ mkdir addTRadImmiscibleIncompressibleTwoPhaseMixture
$ cp -r addTImmiscibleIncompressibleTwoPhaseMixture/
   addTRadImmiscibleIncompressibleTwoPhaseMixture/.
```

```
$ cd addTRadImmiscibleIncompressibleTwoPhaseMixture
$ mv addTImmiscibleIncompressibleTwoPhaseMixture.H
   addTRadImmiscibleIncompressibleTwoPhaseMixture.H
$ mv addTImmiscibleIncompressibleTwoPhaseMixture.C
   addTRadImmiscibleIncompressibleTwoPhaseMixture.C
$ sed -i s/addTImmiscibleIncompressibleTwoPhaseMixture/
   addTRadImmiscibleIncompressibleTwoPhaseMixture/g Make/files
$ sed -i s/addTImmiscibleIncompressibleTwoPhaseMixture/
   addTRadImmiscibleIncompressibleTwoPhaseMixture/g
   addTRadImmiscibleIncompressibleTwoPhaseMixture.*
```

The library path is already set correctly in Make/files to that of the user, and also the Make/options file got the paths corrected hence, everything should be correct, and the process of compiling guarantees it.

```
$ wmake
```

Parameters: T_{env}, ϵ, σ_{SB} and f

Since the radiative term expression is

$$-\frac{\epsilon\sigma_{SB} f A_{fs}}{Vol}\left(T^4 - T_{env}^4\right), \quad [1, -1, -3, 0, 0, 0, 0],$$

we need to build all the pieces that constitutes this term and we start from the parameters such as the emissivity ϵ, the Stephan-Boltzmann constant σ_{SB} ($5.67 \cdot 10^{-8}$ kg s^{-3}K^{-4}), the fractional area of the exposed inner core f, and the environmental temperature T_{env}. The values of those variables are supposed to be given by the user as input parameters, like the value of the surface tension. We add them as private member variables of the transport model class and declare them inside the header file addTRadImmiscibleIncompressibleTwoPhaseMixture.H.

```
class addTRadImmiscibleIncompressibleTwoPhaseMixture
:
    public addTIncompressibleTwoPhaseMixture,
    public interfaceProperties
{

public:
    // Add Rad
    // Private data
        const dimensionedScalar emissivity_;
        const dimensionedScalar sigma_SB_;
        const dimensionedScalar fractionalAreaExposed_;
        const dimensionedScalar T_env_;
    // End add Rad
```

The instruction to initialize those parameters are written inside the function constructor of the class, in addTRadImmiscibleIncompressibleTwoPhaseMixture.C, and in particular the values for those are searched inside the transportPropertiesDict of the test case (as we will see in §A.4.5).

```
Foam::addTRadImmiscibleIncompressibleTwoPhaseMixture::
addTRadImmiscibleIncompressibleTwoPhaseMixture
(
    const volVectorField& U,
    const surfaceScalarField& phi
```

```
)
:
    addTIncompressibleTwoPhaseMixture(U, phi),
    interfaceProperties(alpha1(), U, *this),

    // Add Rad
   emissivity_
    (
        "emissivity", dimensionSet(0,0,0,0,0,0,0), lookup("emissivity")
    ) ,
    sigma_SB_
    (
        "sigma_SB", dimensionSet(1,0,-3,-4,0,0,0), lookup("sigma_SB")
    ) ,
    fractionalAreaExposed_
    (
        "fractionalAreaExposed", dimensionSet(0,0,0,0,0,0,0),
        lookup("fractionalAreaExposed")
    ) ,
    T_env_
    (
        "T_env", dimensionSet(0,0,0,1,0,0,0), lookup("T_env")
    )
    // End add Rad
{}
```

The compiling of the library should ensure the correctness of the modifications done.

```
$ wmake
```

calcSourceRadiation

Finally, we introduce the function `calcSourceRadiation` that computes the radiative heat transfer between the surface of the fluid and the environment. The declaration of this new member function is in the header file, after the declaration of the member function `read`.

```
// Add Rad
// Calculates the radiative source term
tmp<fvScalarMatrix> calcSourceRadiation(
    const volVectorField& U,
    volScalarField& T ,
    volScalarField& RadiativeCoeff
);
// End add Rad
```

The vector-field of velocity `U` is used only to get access at the cell volumes values, while, `T` is the temperature scalar field and `RadiativeCoeff` is a field described later, in §A.4.3, useful for the visualization on ParaView of that part of the domain interested by this thermal exchange.

From the numerical point of view, we need to split this term in two parts, an explicit and an implicit one, and also to linearize the implicit term:

$$-\frac{\epsilon\sigma_{SB} f A_{fs}}{Vol}\left(T^4 - T_{env}^4\right) = \frac{\epsilon\sigma_{SB} f A_{fs}}{Vol} T_{env}^4 - \left(\frac{\epsilon\sigma_{SB} f A_{fs}}{Vol} T^3\right) * T \tag{A.1}$$

The same kind of splitting and linearization is pursued also by the radiation models already implemented in OpenFOAM as thermophysical models. OpenFOAM offers names-

paces of functions that automatically treat source terms: `Foam::fvm::Sp()` for implicit treatment, `Foam::fvm::SuSp()` for implicit/explicit discretization and `Foam::fvm::Su()` for explicit treatment (for more information about the namespaces, the user may consult the Guide) and we discretize the implicit term in Eq. (A.1) exactly with the function `Foam::fvm::Sp()` (in §1.3.2.1 in Eq. (1.133), we introduced the discretization of source terms in explicit and implicit parts).

The function `calcSourceRadiation` is initialized inside the C-file, again after the `read()` function. We use two textttvolScalarField variables for help: `Afs` to host the surface area of the interface cells, and `epsilonMatrix`, a field that is zero everywhere except on the interface where its value is $\epsilon\sigma_{SB} f A_{fs}/Vol$.

```
// Add Rad
Foam::tmp<Foam::fvScalarMatrix> Foam::addTRadImmiscibleIncompressibleTwo
PhaseMixture::calcSourceRadiation(
    const volVectorField& U,
    volScalarField& T ,
    volScalarField& RadiativeCoeff
)
{
    Info << "calcSourceRadiation" << endl;
    // Initialize the radiative coefficient epsilon
    // as 1 or 0 for on interface or not
    volScalarField epsilonMatrix(nearInterface());
    // Initialize the field for the area of the interface surface
    // as 1 or 0 for on interface or not
    volScalarField Afs(epsilonMatrix);

    // Estimate cell surface area as V^(2/3)
    forAll(Afs, cellI) {
        if ( Afs[cellI] > 0 )
          Afs[cellI] = pow(U.mesh().V()[cellI],0.67);
    }
    forAll(epsilonMatrix.boundaryField(), patchI) {
        forAll(epsilonMatrix.boundaryField()[patchI], faceI ) {
            epsilonMatrix.boundaryFieldRef()[patchI][faceI] = scalar
   (0.0);
        }
    }
    // Calculate the volume fraction rate term
    forAll(epsilonMatrix, cellI) {
        if((T[cellI] > T_env_.value())) {
            epsilonMatrix[cellI] = emissivity_.value()
                * fractionalAreaExposed_.value()
                * sigma_SB_.value() * Afs[cellI]
                / U.mesh().V()[cellI] ;
        }else{
            epsilonMatrix[cellI] = scalar(0.0) ;
        }
    }

    dimensionedScalar dimCorr("dimCorr",dimMass/(pow4(dimTemperature)
    *pow3(dimTime)*dimLength),1);
    RadiativeCoeff = epsilonMatrix * dimCorr ;
    return(
        epsilonMatrix * dimCorr * pow4(T_env_)
        - Foam::fvm::Sp( epsilonMatrix * dimCorr * pow3(T) , T)
    );
```

```
}
// End add Rad
```

Since the radiative term is applicable only to the cells belonging to the interface between the two fluids, the function `nearInterface()` helps us in this discernment. Such a function, inherited from the `interfaceProperties` module, returns a `volScalarField` object therefore, when we define a couple of volume scalar fields `A` and `B` as

```
volScalarField  A(nearInterface());
volScalarField  B(A);
```

According to this, the entrances of both `A` and `B` are 1 in correspondence to the interface cells and 0 everywhere else.

Compile the library.

```
$ wmake
```

A.4.2 Editing the solver

Local copy of the new solver

Start by copying the folder interThermalFoam and changing its name to interThermalRadFoam.

```
$ cd $WM_PROJECT_USER_DIR/applications/solvers/multiphase
$ mkdir interThermalRadFoam
$ cp -r interThermalFoam/ interThermalRadFoam/.
$ cd interThermalRadFoam
$ mv interThermalFoam.C interThermalRadFoam.C
```

Rename within the files.

```
$ sed -i s/interThermalFoam/interThermalRadFoam/g Make/files
$ sed -i s/interThermalFoam/interThermalRadFoam/g interThermalRadFoam.C
```

There is no need to change the name of the executable path, because it already coincides with the user bin. Compile to obtain the local version of the `interThermalRadFoam` and to be sure that everything works as expected.

```
$ wmake
```

The new solver has to call the new library:

```
$ sed -i s/addTImmiscibleIncompressibleTwoPhaseMixture/
   addTRadImmiscibleIncompressibleTwoPhaseMixture/g createFields.H
$ sed -i s/addTImmiscibleIncompressibleTwoPhaseMixture/
   addTRadImmiscibleIncompressibleTwoPhaseMixture/g interThermalRadFoam.
   C
$ sed -i s/addTImmiscibleIncompressibleTwoPhaseMixture/
   addTRadImmiscibleIncompressibleTwoPhaseMixture/g Make/options
$ sed -i s/immiscibleIncompressibleTwoPhaseMixture/
   addTRadImmiscibleIncompressibleTwoPhaseMixture/g createFields.H
$ sed -i s/immiscibleIncompressibleTwoPhaseMixture/
   addTRadImmiscibleIncompressibleTwoPhaseMixture/g interThermalRadFoam.
   C
```

The link between the new solver and the new library should be done and the compile of the solver ensure it: compile from inside the folder interThermalRadFoam

```
$ wmake
```

Add the source radiative term to the temperature equation and new field

Inside the file for the temperature equation interThermalRadFoam/TEqn.H, we add the radiation term as a call to the function implemented into our transportModel

```
fvScalarMatrix TEqn
(
  fvm::ddt(rhoCp,T)
    + fvm::div(rhoCpPhi,T)
    - fvm::laplacian(kappaf,T)

    // Add Rad
    ==
    // Radiative term
    mixture.calcSourceRadiation(U,T,RadiativeCoeff)
    // End add Rad

);
```

Moreover, we create the field for `RadiativeCoeff` into createFields.H (which helps to visualize with ParaView where this source term is active.

```
// Add Rad
volScalarField RadiativeCoeff
(
    IOobject
    (
        "RadiativeCoeff",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    mesh,
    dimensionedScalar("RadiativeCoeff",dimMass/(pow4(dimTemperature)
    *pow3(dimTime)*dimLength), 0)
);
// End add Rad
```

A.4.3 Visualization of additional field

We want to visualize two fields involved in the radiative computation. One is `RadiativeCoeff`, already defined, which has great meaning since it should be equal to zero everywhere except at the interface, and then the field describing the temperature dependant part of the radiative term $T^4 - T_{env}^4$ that we call `T4mTenv4`. First of all, we define those fields into the file createFields.H located inside the folder of the solver, i.e. interThermalRadFoam

```
// Add Rad
volScalarField RadiativeCoeff
(
    IOobject
    (
        "RadiativeCoeff",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
```

```
    ),
    mesh,
    dimensionedScalar("RadiativeCoeff",dimMass/(pow4(dimTemperature)*
   pow3(dimTime)*dimLength), 0)
);

volScalarField T4mTenv4
(
    IOobject
    (
         "T4mTenv4",
         runTime.timeName(),
         mesh,
         IOobject::NO_READ,
         IOobject::AUTO_WRITE //NO_WRITE
    ),
    mesh,
    dimensionedScalar("T4mTenv4",pow4(dimTemperature), 0)
);
// end add Rad

#include "createMRF.H"
#include "createFvOptions.H"
```

These fields are both initialized equal to zero everywhere and their values are updated during the simulation and we do that inside the file devoted to temperature equation TEqn.H, in particular, the field `RadiativeCoeff` is defined simultaneously with the call to the function computing the radiative source term `calcSourceRadiation`, while the field `T4mTenv4` is defined separately. Defining the `RadiativeCoeff` field within the function `calcSourceRadiation` optimizes the execution time since that file is computed and used in that function. Instead, the field `T4mTenv4` is not used for the computations, hence we create the new function `calcT4mTenv4` to define it.

RadiativeTerm

We notice that in §A.4.2 the `RadiativeCoeff` field is already passed in input to the function computing the radiative source term, while in §A.4.1 the definition of this field is already written as `RadiativeCoeff = epsilonMatrix*dimCorr`, hence we don't need to write anything else. The function `calcSourceRadiation` receives in input velocity `U`, temperature `T`, and `RadiativeCoeff` field. The velocity field is declared constant because there is no need to change its value, it is only used to detach the cell volumes. The temperature field is not constant because its values need to change since the temperature equation is solved implicitly. Meanwhile, also the `RadiativeCoeff` field is not constant because its values will be updated.

T4mTenv4

The new function `calcT4mTenv4` is declared inside the header file addTRadImmiscibleIncompressibleTwoPhaseMixture.H immediately after the declaration of the member function `calcSourceRadiation`.

```
// Calculates the difference between the fourth powers
// of temperature and environmental temperature
void calcT4mTenv4(
    volScalarField& T4mTenv4
```

```
);
```

In this case the temperature field doesn't change, so it is passed as constant variable. The instructions for this member function are to write inside the file addTRadImmiscibleIncompressibleTwoPhaseMixture.C, added after those of `calcSourceRadiation`.

```
void Foam::addTRadImmiscibleIncompressibleTwoPhaseMixture::calcT4mTenv4(
    const volScalarField& T ,
    volScalarField& T4mTenv4
)
{
    Info << "calcT4mTenv4" << endl;
    forAll(T,cellI){
        T4mTenv4[cellI] = pow4(T[cellI]) - pow4(T_env_.value());
    }
}
```

Notice a small detail: in this case the assignment is done directly on the values of the `volScalarField T4mTenv4`, hence we need to use the dimensionedScalar member function `value()` to only access to the value of `T_env_` without the dimensions, instead in §A.4.1 we wrote `pow4(T_env_)` to consider both the value and the dimension of `T_env_`.

A.4.4 Source term modification to improve the stability

The radiative source term formulation can be slightly modified to improve numerical stability. In Eq. (A.1) we divided and linearized the radiative term. We define the coefficient $\varepsilon := \frac{\epsilon\sigma_{SB} f A_{fs}}{Vol}$, add and subtract $\pm 4\varepsilon T^3 \cdot T$, and rearrange the terms as follows:

$$-\frac{\epsilon\sigma_{SB} f A_{fs}}{Vol}\left(T^4 - T_{env}^4\right) = -\varepsilon\left(T^4 - T_{env}^4\right) \pm 4\varepsilon T^3 \cdot T = \varepsilon\left(T_{env}^4 + 3T^4\right) - 4\varepsilon T^3 * T \quad \text{(A.2)}$$

where we find again an explicit term and a linearized implicit term. This procedure redistribute the source terms and make the coefficient matrix more diagonally dominant. Therefore, we modify the last command of `calcSourceRadiation` (previously described in §A.4.1) into

```
return(
    epsilonMatrix * dimCorr * pow4(T_env_)
    - Foam::fvm::Sp( 4 * epsilonMatrix * dimCorr * pow3(T) , T)
    + epsilonMatrix * dimCorr * 3 * pow4(T)
) ;
```

A.4.5 Run the test Case

The test case of the dam break with temperature, namely the damBreakAddT test of §A.3.3, is modified and to be used again. Inside the file system/controlDict, the call to the function `interThermalFoam` must be substituted with the call to the function `interThermalRadFoam`.

```
$ sed -i s/interThermalFoam/interThermalRadFoam/g system/controlDict
```

Inside the file constant/transportProperties, after the surface tension `sigma`, we add

```
// Add the Radiative transport properties
emissivity              [0 0 0 0 0 0 0]      0.96 ;
```

```
sigma_SB                [1 0 -3 -4 0 0 0]    5.67e-8 ;
fractionalAreaExposed   [0 0 0 0 0 0 0]      1.0 ;
T_env                   [0 0 0 1 0 0 0]     293.0 ;
// End
```

Inside the folder 0, add a file that define the initial conditions for the `RadiativeCoeff` field: copy the temperature file T, change the name as RadiativeCoeff, and then modify it by substituting the body of the listing with:

```
dimensions      [1 -1 -3 -4 0 0 0];
internalField   uniform 0;
boundaryField
{
    ".*"
    {
        type            calculated;
        value           uniform 0;
    }
}
```

Ready to execute the modified dam break test!

```
$ blockMesh
$ setFields
$ interThermalRadFoam | tee -a interThermalRadFoam.log
```

A.5 Add the Convective Heat Loss term

We add a term in the energy equation describing the supposed fluid heat loss due to convection. This term will be active only on the fluid surface, exactly as the radiative term, and even the general implementation is not so far from what was done for the radiative term. There is an additional consideration to do: since we reserve the description of the convective heat loss to a source term, the heat loss due to diffusion through the interface must be suppressed.

The term describing the heat flux depends on the temperature difference between the fluid and the air, it is proportional to the area of the fluid surface A_{fs} m^2, to the heat transfer coefficient λ $\mathrm{kgs}^{-3}\mathrm{K}$, and to the fractional area of the exposed inner core f, in addition, since the energy equation is expressed per unit volume, we must divide by the volume of the cell Vol m^3

$$-\frac{\lambda f A_{fs}}{Vol}(T - T_{env}), \quad [1, -1, -3, 0, 0, 0, 0]. \tag{A.3}$$

We also introduce $\bar{\chi}_{fs}$ as a coefficient for the diffusion term, that prevents the diffusion to verify at the interface between the phases. To add this last coefficient $\bar{\chi}_{fs}$, we edit the file TEqn.H of the energy equation, whereas the convective source term is again implemented within the transport model.

A.5.1 Editing the transport model

The implementation of the convective heat flux term is very similar to that of the radiative term, hence we will create a new transport model class by modifying the previous one and call it `addTRadConvImmiscibleIncompressibleTwoPhaseMixture`.

Local copy of the new transport model

As we have seen in the previous case of the radiative term, the first thing to do is copy the directory of the transport model that we are going to modify and then change all the repetitions of the original class name to the new name.

```
$ cd ~/OpenFOAM/username-v1912/src/transportModels
$ mkdir addTRadConvImmiscibleIncompressibleTwoPhaseMixture
$ cp -r addTRadImmiscibleIncompressibleTwoPhaseMixture/
   addTRadConvImmiscibleIncompressibleTwoPhaseMixture/.
$ cd addTRadConvImmiscibleIncompressibleTwoPhaseMixture
$ mv addTRadImmiscibleIncompressibleTwoPhaseMixture.H
   addTRadConvImmiscibleIncompressibleTwoPhaseMixture.H
$ mv addTRadImmiscibleIncompressibleTwoPhaseMixture.C
   addTRadConvmmiscibleIncompressibleTwoPhaseMixture.C
$ sed -i s/addTRadImmiscibleIncompressibleTwoPhaseMixture/
   addTRadConvImmiscibleIncompressibleTwoPhaseMixture/g Make/files
$ sed -i s/addTRadImmiscibleIncompressibleTwoPhaseMixture/
   addTRadConvImmiscibleIncompressibleTwoPhaseMixture/g
   addTRadConvImmiscibleIncompressibleTwoPhaseMixture.*
```

The path of the library is already correctly set to the user library in Make/files, and even the file Make/options sees the right paths. Compile the class.

```
$ wmake
```

Heat transfer coefficient λ

By observing both the radiative term $-\frac{\epsilon\sigma_{SB}fA_{fs}}{Vol}\left(T^4 - T_{env}^4\right)$ and the convective terms $-\frac{\lambda fA_{fs}}{Vol}\left(T - T_{env}\right)$, one notices that they have in common some parameters, hence we need only to introduce the heat transfer coefficient λ kgs^{-3}K. The value of this parameter is given from the user as input, since it changes according to simulations. It is added as a private member variable of the new transport model. λ is declared inside the header file addTRadConvImmiscibleIncompressibleTwoPhaseMixture.H, together with the other parameters $\epsilon, \sigma_{SB}, T_{env}$, and it is named `heatTransferCoeff`.

```
public:
// Add Rad, add Conv
// Private data
    const dimensionedScalar emissivity_;
    const dimensionedScalar sigma_SB_ ;
    const dimensionedScalar heatTransferCoeff_;
    const dimensionedScalar fractionalAreaExposed_ ;
    const dimensionedScalar T_env_;
// End add Rad, add Conv
```

The initialization of this new parameter is written inside the constructor function of the class inside addTRadConvImmiscibleIncompressibleTwoPhaseMixture.C together with the other parameters, and its value is searched inside the file transportPropertiesDict of the test case.

```
    heatTransferCoeff_
    (
        "heatTransferCoeff", dimensionSet(1,0,-3,-1,0,0,0), lookup("
   heatTransferCoef
        f")
    ) ,
```

A compiling process of the library will ensure the correctness of these few modifications done.

```
$ wmake
```

calcSourceForcedConvection

The function that compute the convective heat loss source term $-\frac{\lambda f A_{fs}}{Vol}(T - T_{env})$ is called `calcSourceForcedConvection`. This new member function of the class is declared in the header file, such as the `calcSourceRadiation` function, after the declaration of the member function `read`.

```
// Add Forced Convection
//- Calculates convective source term, returned as volume fraction rate
tmp<fvScalarMatrix> calcSourceForcedConvection(
    const volVectorField& U,
  volScalarField& T
);
```

`U` is the velocity vector field and we use it to access at the cell volumes values, `T` is the temperature scalar field. From a mathematical point of view, the convective term is linear, conversely from the radiative term (that required a linearization procedure). From a numerical point of view, we need to split it in two parts, one treated explicitly and the other implicitly with the function `Foam::fvm::Sp()` introduced in §A.4.1:

$$-\frac{\lambda f A_{fs}}{Vol}(T - T_{env}) = \frac{\lambda f A_{fs}}{Vol} T_{env} - \frac{\lambda f A_{fs}}{Vol} T.$$

The new function `calcSourceRadiation` is initialized inside the C-file after the `read()` function. Inside the function, we introduce two variables volScalarField: `Afs`, that hosts the area of the fluid surface cells, and `WMatrix` that is zero everywhere except on the interface between the fluids where its value is $\lambda f A_{fs}/Vol$.

```
// Add Conv
Foam::tmp<Foam::fvScalarMatrix>
Foam::addTRadConvImmiscibleIncompressibleTwoPhaseMixture::
    calcSourceForcedConvec
    tion(
    const volVectorField& U,
    volScalarField& T
)
{
    Info << "calcSourceForcedConvection" << endl;
    // Initialize the forced convective coefficient W
    // as 1 or 0 for on interface or not
    volScalarField WMatrix(nearInterface());
    // Initialize the field for the area of the interface surface
    // as 1 or 0 for on interface or not
    volScalarField Afs(WMatrix);

    // Estimate cell surface area as V^(2/3)
    forAll(Afs, cellI) {
        if ( Afs[cellI] > 0 )
            Afs[cellI] = pow(U.mesh().V()[cellI],0.67);
    }
    forAll(WMatrix.boundaryField(), patchI) {
        forAll(WMatrix.boundaryField()[patchI], faceI ) {
```

```
            WMatrix.boundaryFieldRef()[patchI][faceI] = scalar(0.0) ;
        }
    }
    // Calculate the volume fraction rate term
    forAll(WMatrix, cellI) {
        if((T[cellI] > T_env_.value())) {
            WMatrix[cellI] = heatTransferCoeff_.value()
            * fractionalAreaExposed_.value()
            * Afs[cellI] / U.mesh().V()[cellI] ;
        }
        else{
            WMatrix[cellI] = scalar(0.0) ;
        }
    }

    dimensionedScalar dimCorr(
        "dimCorr",dimMass/(dimTemperature*pow3(dimTime)*dimLength),1);
    //   ConvectiveCoeff = WMatrix * dimCorr ;
    /*  activate this in case you create the volScalarField
   ConvectiveCoeff
     * to visualize it on ParaFoam */
    return(
        WMatrix * dimCorr * T_env_
        - Foam::fvm::Sp( WMatrix * dimCorr , T)
    );
}
// End add Conv
```

Compile the library.

```
$ wmake
```

A.5.2 Editing the solver

We create a new solver named `interThermalRadConvFoam` that accounts for both radiative and convective processes, by coping and modifying the solver `interThermalRadFoam` and create the new one, moreover we add the coefficient $\bar{\chi}_{fs}$ for the diffusion term and the function computing the convective source term called `calcSourceForcedConvection`.

Local copy of the new solver

Start by copying the folder interThermalRadFoam and by changing its name in interThermalRadConvFoam.

```
$ cd $WM_PROJECT_USER_DIR/applications/solvers/multiphase
$ mkdir interThermalRadConvFoam
$ cp -r interThermalRadFoam/ interThermalRadConvFoam/.
$ cd interThermalRadConvFoam
$ mv interThermalRadFoam.C interThermalRadConvFoam.C
```

Rename within the files

```
$ sed -i s/interThermalRadFoam/interThermalRadConvFoam/g Make/files
$ sed -i s/interThermalRadFoam/interThermalRadConvFoam/g
   interThermalRadConvFoam.C
```

There is no need to change the name of the executable path, because it already coincides with the user bin. The user can compile it to be sure that everything works as expected.

```
$ wmake
```

The new solver has to call the new library, then, from inside the folder interThermalRadConvFoam set:

```
$ sed -i s/addTRadImmiscibleIncompressibleTwoPhaseMixture/
   addTRadConvImmiscibleIncompressibleTwoPhaseMixture/g createFields.H
$ sed -i s/addTRadImmiscibleIncompressibleTwoPhaseMixture/
   addTRadConvImmiscibleIncompressibleTwoPhaseMixture/g
   interThermalRadFoam.C
$ sed -i s/addTRadImmiscibleIncompressibleTwoPhaseMixture/
   addTRadConvImmiscibleIncompressibleTwoPhaseMixture/g Make/options
$ sed -i s/addTRadImmiscibleIncompressibleTwoPhaseMixture/
   addTRadConvImmiscibleIncompressibleTwoPhaseMixture/g createFields.H
$ sed -i s/addTRadImmiscibleIncompressibleTwoPhaseMixture/
   addTRadConvImmiscibleIncompressibleTwoPhaseMixture/g
   interThermalRadFoam.C
```

The linking between the new solver and the new library should be done correctly. Compile the solver from inside the folder interThermalRadConvFoam.

```
$ wmake
```

Add the convective source term to the temperature equation

Inside the file TEqn.H, the heat equation is modified in order to add the convective source term and the coefficient $\bar{\chi}_{fs}$ for the diffusion term. Start by defining a volScalarField named `NoInterface`, that plays the role of $\bar{\chi}_{fs}$, before the declaration of the `TEqn`.

```
// Add Conv
volScalarField NoInterface ( mixture.nearInterface() );
forAll ( mesh.C(), celli )
{
  if ( NoInterface[celli] > 0)  NoInterface[celli] = 0;
  else
      NoInterface[celli] = 1;
}
// End add Conv
```

Then go through the equation, add the coefficient for the diffusion term, and add the convective source term.

```
fvScalarMatrix TEqn
(
    fvm::ddt(rhoCp,T)
    + fvm::div(rhoCpPhi,T)
    - NoInterface * fvm::laplacian(kappaf,T) // Add the coefficient
    ==
    mixture.calcSourceRadiation(U,T,RadiativeCoeff)
    + mixture.calcSourceForcedConvection(U,T) // Add the convective term
);
```

Compile the solver.

```
$ wmake
```

A.5.3 Run the test Case

Once again, we exploit the test case of the dam break with temperature relying on and modifying the damBreakAddT test of §A.4.5. Inside the file system/controlDict, the call to the function `interThermalRadFoam` must be substituted with the call to the function `interThermalRadConvFoam`.

```
$ sed -i s/interThermalRadFoam/interThermalRadConvFoam/g system/
   controlDict
```

Inside the file constant/transportProperties, after the surface tension `sigma`, we add the `convectiveHeatTransferParameter` together with the radiative parameters already introduced:

```
// Radiative and Convective transport properties
emissivity                      [0 0 0 0 0 0 0]    0.96 ;
sigma_SB                        [1 0 -3 -4 0 0 0] 5.67e-8 ;
fractionalAreaExposed           [0 0 0 0 0 0 0]    1.0 ;
T_env                           [0 0 0 1 0 0 0]    293.0 ;
convectiveHeatTransferParameter  [1 0 -3 -1 0 0 0] 2; // Add
```

Ready to execute the modified dam break test!

```
$ blockMesh
$ setFields
$ interThermalRadConvFoam | tee -a interThermalRadConvFoam.log
```

A.6 Add the Conductive Heat Transfer Boundary Condition

The boundary condition for the conductive heat transfer with the lower wall is added by modifying the file 0/T of the dam break test case damBreakAddT. Start by adding some reference parameters (that must be coherent with those specified in constant/transportProperties):

```
dimensions      [0 0 0 1 0 0 0]; // kg m s K mol A cd
// Add
T_env           293;
fluidDensity    1000;
Pr              10;
cp              4190;
internalField   uniform $T_env;
// End add
```

Modify the boundary condition at the lower wall from `zeroGradient` to what follows:

```
    lowerWall
    {
        type            exprMixed;
        value           $internalField;
        variables       "Tinf=$T_env; rho=$fluidDensity; cp=$cp; DT=1/
   $Pr;
                        k=DT*rho*cp; L_wall=0.002; k_wall=0.03";
        valueExpr       "Tinf";
        fractionExpr    "1.0/(1.0 + ( k/(mag(pos())) * (L_wall/k_wall)
   ))";
    }
```

Execute the program

```
$ blockMesh
$ setFields
$ interThermalRadConvFoam | tee -a interThermalRadConvFoam.log
```

Appendix B

Variational derivation of shallow-water equations

The topics presented in this Appendix have been the subject of the final examination of the reading course "Variational principles in fluid dynamics" taken by Prof. Pinamonti at the University of Genova during the Ph.D. program.

In the first two sections, §B.1 and §B.2, some fundamental notions of calculus of variations are introduced by taking inspirations from Carati and Galgani [31]. Into the third section §B.3 we present the Lagrangian proposed by Luke [177] to derive the water waves equations also showing the explicit computations for the derivation. Luke's Lagrangian is basilar for the works Clamond and Dutykh [45], Clamond and Dutykh [46], in which it is used to derive several models for water waves, among which shallow-water models, by adopting the "relaxed" variational principles. Section §B.4 introduces the concept of relaxation in the context of variational principles and presents the derivation of the relaxed Lagrangian density used by Clamond and Dutykh [45]. The last section §B.5 shows two examples of the modified shallow water systems of equations that are derived from relaxed variational principles by choosing appropriate ansatz and also imposing some constraints. The modified shallow water model derived with the use of relaxed variational principles is different from our modified model derived in §2.1.

B.1 Introduction to the variational principles

In this section, we introduce the Hamilton Variational Principle for a mechanical system, also referred to as *Principle of least action* or, more accurately, as *Principle of stationary action*. The Hamilton Principle is essentially a substitute of the more familiar mechanical principle which characterizes the "physical" movements as those satisfying the Newton equations or the Lagrange equations. As it usually happens, the new point of view may result to be more convenient for "heuristic" aims, namely when one wants to extend old theories to new fields.

Variational principles became a central discussion argument inside the scientific community one century before Hamilton's works (dated half '800) and were accompanied by considerable attention to their philosophical implications; their official birth year might date 1744. Sometime earlier, a lot of discussions arose from the formulation given by Maupertuis about the Fermat Principle for the geometric optics [1] and from his transpo-

[1] In optics, *Fermat's principle* or the *principle of least time*, named after the French mathematician

sition in purely mechanical formulation. Later, Euler brought back the problem to the mathematical point of view stimulated by the Bernoulli discussion about the brachistochrone problem [2]. Euler formulated the problem of analytically characterizing the *geodetics* (i.e. the minimum length curves on an assigned surface) and published the solution to this problem inside the work of 1744 titled as "*Methudus inveniendi lineas curvas maximi minive proprietates gaudentes*". Furthermore, such a work contains the solution of a more general problem (that is the core of this section), that is to characterize as solutions of differential equations the functions that are stationary points of an assigned integral functional (the meaning of these words are clarified in the following). The method used nowadays to discuss the variational principles is substantially the one introduced by Lagrange (1736–1813) in his first work, which gave him notoriety. Hamilton added his contribution in three works: "*On a general method of expressing the* paths of light and the planets, *by the coefficients of a characteristic function*" [3], 1833, "*On a general method in dynamics*", 1834, "*Second essay on a general method in dynamics*", 1835.

From the Newtonian or Lagrangian point of view, a movement or a trajectory is described by differential equations, whereas the fundamental idea at the basis of variational principles is to characterize the movement or the trajectory by the property of being the minimum or maximum with respect to a family of movements or trajectories. The most obvious example, thinking about trajectories, is the straight line on a plane. In fact, on one hand, a straight line $y = y(x)$ satisfies the differential equation $y'' = 0$ because it has null curvature, and, on the other hand, it is "represents" the minimum length curve: fixed two points A and B on the plane and considered the set of all the curves that connect the two points, the straight segment $\overline{AB}$ (laying on $y(x)$) is the curve of minimum length connecting A and B, see Figure B.1.

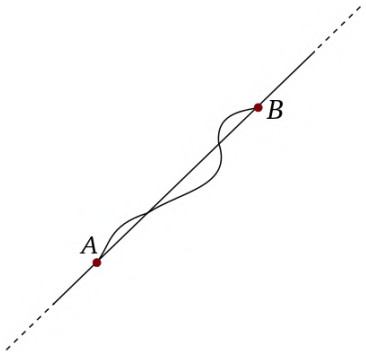


Figure B.1: The straight line passing trough two points is the minimum length curve.

From a wider point of view, variational principles provide a **global** characterization instead of a **local** characterization. In such a sense, the straight line characterization,

Pierre de Fermat, is the principle stating that the path taken between two points by a ray of light is the path traversed in the least time. This principle is sometimes taken as the definition of a ray of light.

[2] In mathematics and physics, a *brachistochrone* curve (from Ancient Greek *brákhistos khrónos* means 'shortest time'), or curve of fastest descent. Such a curve, lying on a plane, connects a point A to a lower point B, where B is not directly below A, and a bead slides frictionlessly under the influence of a uniform gravitational field to a given endpoint in the shortest time. The brachistochrone problem solution is not a straight line nor a combination thereof, but a cycloid.

[3] The reference to *light and planets*, and so the correspondence between *waves and corpuscles* become later the fulcrum for the passage from the classic to the quantum mechanics with the Schroedinger procedure.

as that curve that minimizes the distance between two points on a plane, is a global property; in addition, this characterization involves the length notion, a property regarding the whole curve. Otherwise, the requirement that the second derivative y'' vanishes is applied for every single point x and, for this reason, the characterization with the differential equation is seen as the union of local properties.

Another relevant aspect that differentiates the global and local characterization refers to the property of invariance with respect to the coordinates. In fact, a curve may be described analytically in different ways according to the chosen coordinates, therefore also the characterization of local properties depends on the coordinates chosen. Instead, the global properties, typically expressed by indefinite integrals, are independent of the coordinates, and, for this reason, the global characterizations are more significant. We could say that Newton or Lagrange (or Hamilton) equations constitute the local form of the variational principle (of global type) that is the *Hamilton variational principle*. In the next, the variational formulation of the dynamic is introduced.

B.1.1 Variational formulation of Lagrangian mechanics

Speaking about the mathematical environment in which we move, we consider a set of curves $\mathcal{U}$ accompanied by a *functional* F, $F : \mathcal{U} \to \mathbb{R}$, which in the previous example corresponds to the functional that associates each curve to its length between A and B. The aim is to study the existence of possible local minima of the functional F.

We start by considering a Lagrangian system, that means a particular class of systems of differential equations whose form is completely determined from a scalar function. We fix a time interval $I = [t_1, t_2] \subset \mathbb{R}$ and consider Ω an open set of $\mathbb{R}^n$ named **space of configurations** of dimension n, in particular Ω is a variety and a point on Ω is individuated by the coordinates $\mathbf{q} = (q_1, \ldots, q_n) \in \mathbb{R}^n$ called **Lagrangian coordinates**. We introduce the function

$$\begin{aligned} L : I \times \Omega \times \mathbb{R}^n &\longrightarrow \mathbb{R}, \\ (t,\ \mathbf{q},\ \dot{\mathbf{q}}) &\longrightarrow L(t, \mathbf{q}(t), \dot{\mathbf{q}}(t)) \end{aligned}$$

that is a function in C^2 named **Lagrangian function**. The dimension n indicates the degrees of freedom.

Two Lagrangian functions are said **equivalent** if they lead to the same system of equations. Trivially, in the case they differ of a constant, then they are equivalent. In general, given a function $f : \mathbb{R} \times \Omega \to \mathbb{R}$, $f \in C^2$, the Lagrangian L is equivalent to

$$\bar{L}(t, \mathbf{q}, \dot{\mathbf{q}}) = L(t, \mathbf{q}, \dot{\mathbf{q}}) + \frac{d}{dt} f(t, \mathbf{q}). \tag{B.1}$$

We consider the movements $\mathbf{q} = \mathbf{q}(t)$ that takes place into a fixed interval of time $[t_1, t_2]$ and that satisfy certain boundary conditions, for example those with fixed extremes $\mathbf{q}(t_1) = \mathbf{q}_1$, $\mathbf{q}(t_2) = \mathbf{q}_2$ as represented in Figure B.2, so the functional space $\mathcal{U}$ is defined by

$$\mathcal{U} = \{\mathbf{q} = \mathbf{q}(t) : t_1 \leqslant t \leqslant t_2,\ \mathbf{q}(t_1) = \mathbf{q}_1,\ \mathbf{q}(t_2) = \mathbf{q}_2\}. \tag{B.2}$$

We define the functional of the **Hamiltonian action** $\mathcal{L} : \mathcal{U} \to \mathbb{R}$ as follows

$$\mathcal{L} := \int_{t_1}^{t_2} L(t, \mathbf{q}, \dot{\mathbf{q}})\, dt, \tag{B.3}$$

where the integrand function is called **density** or **Lagrangian density**, and we formulate the principle

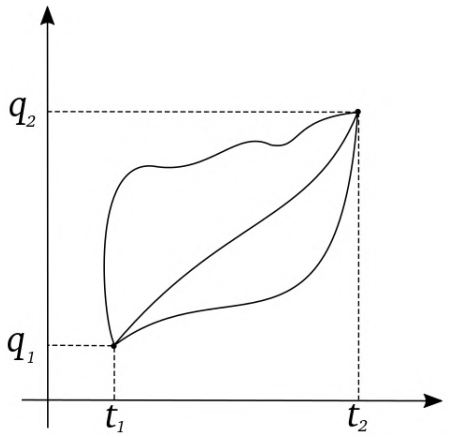


Figure B.2: Domain of admissible functions.

Hamilton's Principle: For a Lagrangian system with $L(t, \mathbf{q}, \dot{\mathbf{q}})$, the natural motions are the stationary points (which are functions) of the Hamiltonian action $\mathcal{L}$, i.e. they are the movements $\mathbf{q} = \mathbf{q}(t)$ such that

$$\delta\mathcal{L} = \delta \int_{t_1}^{t_2} L(t, \mathbf{q}, \dot{\mathbf{q}})\, dt = 0$$

where by δ we denote the analogue of the differential operator for the functional spaces (we discuss it briefly in the following).

The importance of this principle relies on the theorem:

Theorem B.1. *The movements $\boldsymbol{q} = \boldsymbol{q}(t)$ with domain defined in Eq. (B.2) that are the stationary points of the Hamiltonian action $\mathcal{L}$ of Eq. (B.3) are all and only those that satisfy the Lagrange equations*

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}} - \frac{\partial L}{\partial \mathbf{q}} = 0$$

and the assigned boundary conditions.

Thanks to this theorem, Hamilton's principle is considered the fundamental principle of mechanics.

B.2 Notes on calculus of variations

We spend some words speaking about the calculus of variations and, in particular, we see the extension of differential and derivative notions to the functional spaces, which is a classic topic of functional analysis. In order to extend the notion of differential for functionals with domain in functional space, we recall known notions for functions with domain in $\mathbb{R}^n$.

Given a functional $F : \mathbb{R}^n \to \mathbb{R}$, we remind that it has a minimum or a maximum in $\mathbf{x} \in \mathbb{R}^n$ if and only if $\mathbf{x}$ is a stationary point, and this condition coincides with a null differential $dF(\mathbf{x}) = 0$, which, in turn, means that all the partial derivatives of F must be equal to zero, $\dfrac{\partial F}{\partial x_i}(\mathbf{x}) = 0$, $i = 1, \ldots, n$. In order to define the differential of F, we consider the increment of the functional when we move from a point $\mathbf{x}$ to a point $\mathbf{x} + \mathbf{h}$

and we study how the increment of F depends on the movement $\mathbf{h}$. The functional F is differentiable if we have

$$F(\mathbf{x}+\mathbf{h}) = F(\mathbf{x}) + A(\mathbf{x})\mathbf{h} + R,$$

where A is a linear operator and R a rest of the second-order (or even more) with respect to the increment $\mathbf{h}$; the part of the expression that is linear with respect to the increment is called *differential*. From a formal point of view, this definition can be easily extended to the case in which the domain is a functional space.

We consider, as example, the Hamiltonian action $\mathcal{L} = \int_{t_1}^{t_2} L(t, \mathbf{q}, \dot{\mathbf{q}})\, dt$ defined on the functional space $\mathcal{U}$ of Eq. (B.2). The increment of $\mathcal{L}$ is evaluated by passing from the "point" $\mathbf{q}(t)$ to another curve $(\mathbf{q}+\mathbf{h})(t)$, with $\mathbf{h} \in C^\infty([t_1, t_2])$:

$$\mathcal{L}(\mathbf{q}+\mathbf{h}) - \mathcal{L}(\mathbf{q}) = \int_{t_1}^{t_2} \left(L(t, \mathbf{q}+\mathbf{h}, \dot{\mathbf{q}}+\dot{\mathbf{h}}) - L(\mathbf{q}, \dot{\mathbf{q}}, t)\right) dt. \tag{B.4}$$

By using the Taylor expansion at the first order of the functional L, it descends that

$$L(t,\, \mathbf{q}+\mathbf{h},\, \dot{\mathbf{q}}+\dot{\mathbf{h}}) - L(t,\, \mathbf{q},\, \dot{\mathbf{q}}) = \frac{\partial L}{\partial \mathbf{q}}\,\mathbf{h} + \frac{\partial L}{\partial \dot{\mathbf{q}}}\,\dot{\mathbf{h}} \;+\; R \tag{B.5}$$

where R is the rest of the second order (or even more) with respect to the norm of the increments $\mathbf{h}$ and $\dot{\mathbf{h}}$; in addiction we can rearrange the second term on the RHS as follows

$$\frac{\partial L}{\partial \dot{\mathbf{q}}}\,\dot{\mathbf{h}} \;=\; \frac{\partial L}{\partial \dot{\mathbf{q}}}\,\frac{d\mathbf{h}}{dt} \;=\; \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\mathbf{q}}}\,\mathbf{h}\right) - \left(\frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}}\right)\mathbf{h}. \tag{B.6}$$

By introducing the results of Eqs. (B.5,B.6) inside Eq. (B.4), we find that

$$\begin{aligned}\mathcal{L}(\mathbf{q}+\mathbf{h}) - \mathcal{L}(\mathbf{q}) &= \int_{t_1}^{t_2} \left[\frac{\partial L}{\partial \mathbf{q}}\,\mathbf{h} + \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\mathbf{q}}}\,\mathbf{h}\right) - \left(\frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}}\right)\mathbf{h} + R\right] dt \\ &= \int_{t_1}^{t_2} \left(\frac{\partial L}{\partial \mathbf{q}} - \frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}}\right)\mathbf{h}\, dt \;+ \frac{\partial L}{\partial \dot{\mathbf{q}}}\,\mathbf{h}\Big|_{t_1}^{t_2} \;+\; R\end{aligned}$$

therefore, the increment of the functional $\mathcal{L}$ is constituted of the sum of a part that linearly depends on the increment $\mathbf{h}$, and of a part that is the rest R of order higher than one; in particular, the linear part is the sum of two terms: an integral term and a boundary term, i.e. a term depending on the boundary values of $\mathbf{h}$. Finally, the differential of $\mathcal{L}$ is defined, for analogy, as the linear part of its increment. Traditionally, when the domain is a functional space the differential of a functional is denoted by the symbol δ instead of d and it is called **variation**, therefore we will speak about the **variation** $\delta\mathcal{L}$ of the functional $\mathcal{L}$; coherently, also the increment $\mathbf{h}$ of the point $\mathbf{q}$ will be denoted as $\delta\mathbf{q}(t)$ and called **variation** of $\mathbf{q}$.

In conclusion, we saw that the functional of Hamiltonian's action $\mathcal{L} = \int_{t_1}^{t_2} L(t, \mathbf{q}, \dot{\mathbf{q}})\, dt$, admits differential (classic term: *variation*) $\delta\mathcal{L}$ which expression is:

$$\delta\mathcal{L} = \int_{t_1}^{t_2} \left(\frac{\partial L}{\partial \mathbf{q}} - \frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}}\right)\delta\mathbf{q}\, dt \;+ \mathbf{p}\,\delta\mathbf{q}\Big|_{t_1}^{t_2} \tag{B.7}$$

where we have used the definition of momentum $\mathbf{p} = \frac{\partial L}{\partial \dot{\mathbf{q}}}$. Furthermore, we observe that the quantities $\mathbf{q}(t)$ and $\delta\mathbf{q}(t)$ are vectors in $\mathbb{R}^n$, therefore the integral inside Eq. (B.7) must be intended as follows:

$$\int_{t_1}^{t_2} \left(\frac{\partial L}{\partial \mathbf{q}} - \frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}}\right)\delta\mathbf{q}\, dt = \int_{t_1}^{t_2} \sum_{i=0}^{n} \left(\frac{\partial L}{\partial q_i} - \frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i}\right)\delta q_i\, dt.$$

The *stationary point* notion is introduced as analog to the finite-dimensional case.

Definition B.1. *Considering the functional* $\mathcal{L} = \int_{t_1}^{t_2} L(t, \boldsymbol{q}, \dot{\boldsymbol{q}})\, dt$, *defined on the domain* $\mathcal{U}$ *(B.2), a point* $\mathbf{q} = \mathbf{q}(t)$ *is a* **stationary point** *of* $\mathcal{L}$ *if the differential (*variation*) of* $\mathcal{L}$ *is null in this point, namely when*

$$\delta\mathcal{L}(\mathbf{q}) = 0.$$

We also introduce a major Theorem that characterizes the stationary points of $\mathcal{L}$.

Theorem B.2. *The stationary points* $\mathbf{q}(t)$ *of* $\mathcal{L}$ *are characterized by the following independent properties:*

i) they satisfy the differential equation (called Euler-Lagrange equation)

$$\frac{\partial L}{\partial \mathbf{q}} - \frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}} = 0;$$

ii) at the edges of the time interval considered, they satisfy the condition

$$\mathbf{p}(t_1)\, \delta\mathbf{q}(t_1) = \mathbf{p}(t_2)\, \delta\mathbf{q}(t_2)$$

for arbitrary variations $\delta\mathbf{q}(t_1)$, $\delta\mathbf{q}(t_2)$ *according to the temporal boundary conditions assigned in* t_1 *and* t_2.

The second property introduced is generally used to determine, among all the possible solutions of the Euler-Lagrange equation, the one that is stationary. The simplest situation that one may find is that one in which the temporal boundary conditions are assigned, for example $\mathbf{q}(t_1) = \mathbf{q}_1$, $\mathbf{q}(t_2) = \mathbf{q}_2$, indeed in this case the boundary conditions on the variations become trivial: $\delta\mathbf{q}(t_1) = \delta\mathbf{q}(t_2) = 0$.

Notice that the Hamilton principle extends the theory of the Lagrange equation because the function $\mathbf{q}(t)$ can be piece-wise C^1 to make the functional of action well defined, whereas the function $\mathbf{q}(t)$ must be C^2 in order to define the Lagrange equation.

B.3 Luke's Lagrangian for water waves

Our interest is to apply the previous ideas to describe the dynamics of water waves with a particular focus on the shallow water approximation. First of all, we report the equations of motion for irrotational, incompressible, and inviscid fluids we refer to. For simplicity, we consider a 2 dimensional channeled motion of an inviscid fluid (see Figure B.3); the bottom is described by the function $B = B(x, t)$ and the free surface is expressed by the function $h = h(x, t)$. Fixed a spatial point (x, z), velocity is $\mathbf{u} = (u, v)$, where $u, v : \mathbb{R}^3 \longrightarrow \mathbb{R}$ (since $\mathbf{u} = \mathbf{u}(x, z, t)$). The fluid density is denoted by ρ as usual.

The classical mathematical formulation of surface gravity waves involves five equations (since this topic is standard and well known, we report only the equations, but the read can find details in Whitham [272, Chapter 13]). The assumptions on the physical properties of the fluid determine the motion equations:

(i) we consider an *irrotational* fluid then it exists the velocity potential $\Phi(t, x, z)$, that is a primitive for the differential form associated with $\mathbf{u}$, such that $\mathbf{u} = \nabla\Phi$; therefore the irrotational condition expresses as follows

$$\nabla \times \nabla\Phi = 0; \tag{B.8}$$

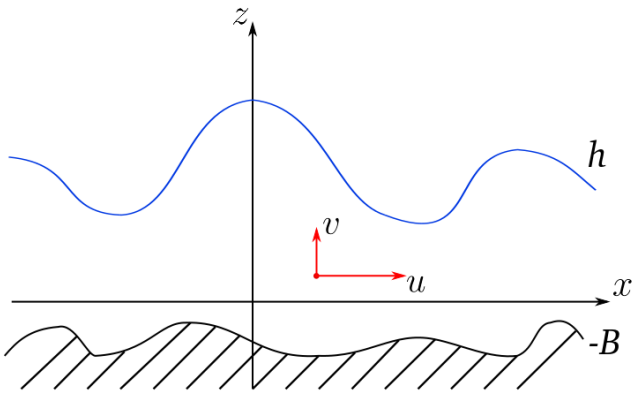


Figure B.3: Example of the situation and main notations used in the discussion.

(*ii*) the *incompressible* fluid condition, which translates the mass conservation equation into the kinematic constraint $\nabla \cdot \mathbf{u} = 0$, writes in terms of velocity potential as follows:

$$\nabla^2 \Phi = \Delta \Phi = 0; \tag{B.9}$$

(*iii*) the Euler equation for the momentum conservation of an irrotational, incompressible, and inviscid fluid writes as follows

$$\Phi_t + \frac{1}{2}\left(\Phi_x^2 + \Phi_z^2\right) + \frac{p - p_0}{\rho} + gz = 0, \tag{B.10}$$

(where we are neglecting the surface tension) and this equation also represents the Bernoulli equation for unsteady potential flow, where the scalar p is the mechanical pressure of water, p_0 is the pressure in the undisturbed air (which we can take to be zero), and g is the gravity acceleration;

(*iv*) there are two boundary conditions at the free surface $z = h$: one is a *kinematic* condition that imposes the fluid to not cross the free surface $h(x,t)$, like a sort of "impermeability" condition, for whom the velocity of the fluid normal to the free-surface must be equal to the velocity of the free-surface normal to itself, and the other condition is a *dynamic* boundary condition of "isobarity" for which the pressure in the water must coincide with the pressure of air at the free surface, namely the condition $p = p_0$ must be verified at the free surface $z = h$:

$$\Phi_z = h_t + \Phi_x \, h_x, \tag{B.11}$$

$$\Phi_t + \frac{1}{2}\left(\Phi_x^2 + \Phi_z^2\right) + gh = 0; \tag{B.12}$$

(*v*) an "impermeability" free-slip boundary condition is set also at the bottom $z = -B(x,t)$ for which the component of the fluid velocity normal to the bottom must vanish:

$$\Phi_z + B_t + \Phi_x \, B_x = 0, \tag{B.13}$$

and the term B_t is null when bathymetry does not change over time.

Analytic solutions of the system of Eqs. (B.8–B.13) (in the unknowns Φ, p) are for special cases, as observed by Peregrine [211]. From a historical point of view, the

water waves theory was developed by constructing several approximations, for example, linearized equations, shallow water, finite depth, and deep water approximations. In the framework of the variational principles for water waves, there are two variational formulations for irrotational surface waves that are commonly used: the *Lagrangian* of Luke [177] and the *Hamiltonian* of Petrov [214] and Zakharov [283]; we are interested in the first one and we introduce it in the following.

The general problem of finding suitable Lagrangian functions does not have a satisfactory solution. The traditional form of Lagrangian for the water wave problem is equal to kinetic minus the potential energy

$$L^* = \int_{-B}^{h} \left[\frac{1}{2} \left(\Phi_x^2 + \Phi_z^2 \right) - gz \right] dz.$$

Luke [177] proposed a definition of Lagrangian as *density* of the functional of the Hamiltonian action (consider the analogy with the definition (B.3)) by using pressure from Eq. (B.10), which means that the principle is one of stationary pressure:

$$\mathcal{L} = \int_{t_1}^{t_2} \int_{x_1}^{x_2} \rho L \, dx \, dt, \quad L = -\int_{-B}^{h} \left[\frac{1}{2} \left(\Phi_x^2 + \Phi_z^2 \right) + \Phi_t + gz \right] dz. \tag{B.14}$$

We prove that, by using the variational principles, it is possible to derive the five equations of motion (B.8)–(B.13) with $\mathcal{L}$ defined in Eq. (B.14).

Notice that $\mathcal{L}$ in Eq. (B.14) is a functional depending on four different variables, namely $\mathcal{L} = \mathcal{L}(\Phi, \rho, h, B)$. For simplicity we define the functional spaces in which we consider the variations; assuming that $\Omega := [t_1, t_2] \times [x_1, x_2]$, then:

$$\mathcal{V} := \left\{ \phi \in C_C^\infty(\Omega) : \phi|_{\partial\Omega} = 0 \right\}, \tag{B.15}$$

$$\mathcal{U} := \left\{ \phi \in C_C^\infty(\Omega \times [-B(x,t), h(x,t)]) : \phi|_{\partial\Omega} = 0 \right\}. \tag{B.16}$$

where C_C^∞ denotes the class of C^∞ functions with compact support.

Lastly, since the variations are present even inside the extremes of integration, we report the useful **Leibniz rule** for the derivative of an integral function:

$$\frac{d}{dx} \int_{\alpha(x)}^{\beta(x)} f(x,t) \, dt = \frac{d\beta}{dx} f(x, \beta(x)) - \frac{d\alpha}{dx} f(x, \alpha(x)) + \int_{\alpha(x)}^{\beta(x)} \frac{\partial}{\partial x} f(x,t) \, dt. \tag{B.17}$$

In order to have well-ordered and simple passages of computations, we consider the variations of the variables one at a time.

Variations of ρ. We consider a variation of density ρ and define $\rho_\varepsilon := \rho + \varepsilon \, \delta\rho$, where $\delta\rho \in \mathcal{U}$, see Eq. (B.16), and $\varepsilon > 0$, then the expression of the functional of the Hamilton action (which is defined in Eq. (B.14)) in such a point follows:

$$\begin{aligned}
\mathcal{L}(\Phi, \rho_\varepsilon, h, B) &= \int_{t_1}^{t_2} \int_{x_1}^{x_2} (\rho + \varepsilon \, \delta\rho) \int_{-B}^{h} \left[\frac{1}{2} \left(\Phi_x^2 + \Phi_z^2 \right) + \Phi_t + gz \right] dz \, dx \, dt \\
&= \int_{t_1}^{t_2} \int_{x_1}^{x_2} \rho \int_{-B}^{h} \left[\frac{1}{2} \left(\Phi_x^2 + \Phi_z^2 \right) + \Phi_t + gz \right] dz \, dx \, dt \\
&\quad + \int_{t_1}^{t_2} \int_{x_1}^{x_2} \varepsilon \, \delta\rho \int_{-B}^{h} \left[\frac{1}{2} \left(\Phi_x^2 + \Phi_z^2 \right) + \Phi_t + gz \right] dz \, dx \, dt.
\end{aligned}$$

The limit of the quotient between the variation of $\mathcal{L}$ with respect to the variation ε is:

$$\lim_{\varepsilon\to 0}\frac{\mathcal{L}(\rho_\varepsilon)-\mathcal{L}(\rho)}{\varepsilon}=\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h}\delta\rho\left[\frac{1}{2}\left(\Phi_x^2+\Phi_z^2\right)\,+\,\Phi_t\,+\,gz\right]dz\,dx\,dt,$$

so we search for the point (Φ,ρ,h,B) which is stationary for the variation $\delta\rho\in\mathcal{U}$. Therefore, by the Fundamental Lemma of calculus of variations [100, Lemma 1, page 9] we have that

$$\delta\mathcal{L}\,(\Phi,\rho,h,B)=0,\quad\forall\,\delta\rho\in\mathcal{V}\quad\Longleftrightarrow\quad\frac{1}{2}\left(\Phi_x^2+\Phi_z^2\right)\,+\,\Phi_t\,+\,gz=0$$

then we find that the Euler equation for non-viscous fluid, that is Eq. (B.10), except for the pressure term, identifies a stationary point of $\mathcal{L}$.

For the sake of simplicity, from here on, the fluid density ρ is assumed to be constant and can thus be set equal to unity without any loss of generality; this is not a limitation for our purpose.

Variations of h. We consider a variation of the function h that represents the free-surface height defining it as $h_\varepsilon:=h+\varepsilon\,\delta h$, with $h\in\mathcal{V}$ (B.15) and $\varepsilon>0$; then the expression of the functional $\mathcal{L}$ in the point (Φ,h_ε,B) is

$$\begin{aligned}\mathcal{L}(\Phi,h_\varepsilon,B)&=\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h+\varepsilon\,\delta h}\left[\frac{1}{2}\left(\Phi_x^2+\Phi_z^2\right)\,+\,\Phi_t\,+\,gz\right]dz\,dx\,dt\\&=\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h+\varepsilon\,\delta h}\frac{1}{2}\Phi_x^2\,dz\,dx\,dt+\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h+\varepsilon\,\delta h}\frac{1}{2}\Phi_z^2\,dz\,dx\,dt\\&\quad+\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h+\varepsilon\,\delta h}\Phi_t\,dz\,dx\,dt+\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h+\varepsilon\,\delta h}gz\,dz\,dx\,dt\end{aligned}$$

We derive the functional with respect to the variation. Since the computations are similar for each of the four terms written on the right-hand side of the previous equation, we see the details only of the derivation of the first term:

$$\begin{aligned}\frac{d}{d\varepsilon}\left(\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h+\varepsilon\,\delta h}\frac{1}{2}\Phi_x^2\,dz\,dx\,dt\right)\Bigg|_{\varepsilon=0}&=\int_{t_1}^{t_2}\int_{x_1}^{x_2}\frac{d}{d\varepsilon}\left(\int_{-B}^{h+\varepsilon\,\delta h}\frac{1}{2}\Phi_x^2\,dz\right)\Bigg|_{\varepsilon=0}dx\,dt\\&\overset{(B.17)}{=}\int_{t_1}^{t_2}\int_{x_1}^{x_2}\left(\delta h\,\frac{1}{2}\big(\Phi_x(h+\varepsilon\,\delta h)\big)^2\right)\Bigg|_{\varepsilon=0}dx\,dt\\&=\int_{t_1}^{t_2}\int_{x_1}^{x_2}\delta h\,\frac{1}{2}\big(\Phi_x(h)\big)^2\,dx\,dt.\end{aligned}$$

The final result of the derivative of the four terms turns out to be

$$\frac{d}{d\varepsilon}\mathcal{L}(\Phi,h_\varepsilon,B)\Big|_{\varepsilon=0}=\int_{t_1}^{t_2}\int_{x_1}^{x_2}\delta h\,\left\{\frac{1}{2}\Big[\big(\Phi_x(h)\big)^2+\big(\Phi_z(h)\big)^2\Big]+\Phi_t(h)+gh\right\}dx\,dt$$

and, for the Fundamental Lemma of the calculus of variations, this is equal to zero for any variation $\delta h\in\mathcal{V}$ if and only if the point (Φ,h,B) verifies the dynamic boundary condition of Eq. (B.12):

$$\frac{d}{d\varepsilon}\mathcal{L}(\Phi,h_\varepsilon,B)\Big|_{\varepsilon=0}=0,\ \forall\,\delta h\in\mathcal{C}\iff\frac{1}{2}\Big[\big(\Phi_x(h)\big)^2+\big(\Phi_z(h)\big)^2\Big]+\Phi_t(h)+gh=0.$$

Variations of Φ. Finally, we consider the variations of Φ so define $\Phi_\varepsilon = \Phi + \varepsilon\delta\Phi$, where $\delta\Phi \in \mathcal{U}$ (B.16) and $\varepsilon > 0$. The functional $\mathcal{L}$ is derived with respect to the variation of Φ obtaining the following expression:

$$
\begin{aligned}
\frac{d}{d\varepsilon}\mathcal{L}(\Phi_\varepsilon, h, B)\Big|_{\varepsilon=0} &= \frac{d}{d\varepsilon}\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h}\Big\{\frac{1}{2}\Big[(\Phi+\varepsilon\delta\Phi)_x^2 + (\Phi+\varepsilon\delta\Phi)_z^2\Big] + (\Phi+\varepsilon\delta\Phi)_t \\
&\quad + gz\Big\}dz\,dx\,dt\Big|_{\varepsilon=0} \\
&= \frac{d}{d\varepsilon}\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h}\Big\{\frac{1}{2}\Big[\Phi_x^2 + 2\,\varepsilon\,\Phi_x(\delta\Phi)_x + \varepsilon^2(\delta\Phi)_x^2 + \Phi_z^2 + 2\,\varepsilon\,\Phi_z(\delta\Phi)_z \\
&\quad + \varepsilon^2(\delta\Phi)_z^2\Big] + \Phi_t + \varepsilon(\delta\Phi)_t + gz\Big\}dz\,dx\,dt\Big|_{\varepsilon=0} \\
&= \int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h}\Big\{\frac{1}{\cancel{2}}\Big[\cancel{2}\,\Phi_x(\delta\Phi)_x + \cancel{2}\,\varepsilon(\delta\Phi)_x^2 + \cancel{2}\,\Phi_z(\delta\Phi)_z + \cancel{2}\,\varepsilon(\delta\Phi)_z^2\Big] \\
&\quad + (\delta\Phi)_t\Big\}dz\,dx\,dt\Big|_{\varepsilon=0} \\
&= \int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h}\Big\{\Phi_x(\delta\Phi)_x + \Phi_z(\delta\Phi)_z + (\delta\Phi)_t\Big\}dz\,dx\,dt \\
&= \underbrace{\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h}\Phi_x(\delta\Phi)_x\,dz\,dx\,dt}_{\text{(A)}} + \underbrace{\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h}\Phi_z(\delta\Phi)_z dz\,dx\,dt}_{\text{(B)}} \\
&\quad + \underbrace{\int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h}(\delta\Phi)_t\,dz\,dx\,dt.}_{\text{(C)}}
\end{aligned}
$$

We start from computing the term (A). At the second passage of the following claim, indicated with $(*)$, we introduce the *Step Function* also called *Heaviside Function* (whose value is zero for negative argument and one for positive argument) in order to change the integral interval, whereas at $(**)$ we use the hypothesis that the variation $\delta\Phi \in \mathcal{U}$ (B.16) is null at x_1 and x_2; we also highlight that we denote by $\delta(z)$ the well known *Dirac Delta function*.

$$
\begin{aligned}
\text{(A)} &\overset{(def.)}{=} \int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h}\frac{\partial\Phi}{\partial x}\frac{\partial(\delta\Phi)}{\partial x}\,dz\,dx\,dt \\
&\overset{(*)}{=} \int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{\mathbb{R}}\frac{\partial\Phi}{\partial x}\frac{\partial(\delta\Phi)}{\partial x}\,\theta(z-(-B))\,\theta(h-z)\,dz\,dx\,dt \\
&= \int_{t_1}^{t_2}\int_{\mathbb{R}}\int_{x_1}^{x_2}\underbrace{\frac{\partial(\delta\Phi)}{\partial x}}_{f}\underbrace{\frac{\partial\Phi}{\partial x}\Big(\theta(z+B)\,\theta(h-z)\Big)}_{G}\,dx\,dz\,dt \\
&\overset{(int.by\,parts)}{=} \int_{t_1}^{t_2}\int_{\mathbb{R}}\Big\{\underbrace{\delta\Phi}_{F}\underbrace{\frac{\partial\Phi}{\partial x}\Big(\theta(z+B)\,\theta(h-z)\Big)}_{G}\Big|_{x_1}^{x_2} +
\end{aligned}
$$

$$
\begin{aligned}
&- \int_{x_1}^{x_2} \underbrace{\delta\Phi}_{F} \underbrace{\left[\frac{\partial^2 \Phi}{\partial x^2} \Big(\theta(z+B)\,\theta(h-z) \Big) + \frac{\partial \Phi}{\partial x} \frac{\partial}{\partial x} \Big(\theta(z+B)\,\theta(h-z) \Big) \right]}_{g} dx \Bigg\} dz\,dt \\
&\overset{(**)}{=} - \int_{t_1}^{t_2} \int_{\mathbb{R}} \int_{x_1}^{x_2} \delta\Phi \frac{\partial^2 \Phi}{\partial x^2} \Big[\theta(z+B)\,\theta(h-z) \Big] dx\,dz\,dt + \\
&\quad - \int_{t_1}^{t_2} \int_{\mathbb{R}} \int_{x_1}^{x_2} \delta\Phi \frac{\partial \Phi}{\partial x} \left[\delta(z+B) \frac{\partial B}{\partial x} + \delta(h-z) \frac{\partial h}{\partial x} \right] dx\,dz\,dt \\
&= - \int_{t_1}^{t_2} \int_{x_1}^{x_2} \int_{-B}^{h} \delta\Phi \frac{\partial^2 \Phi}{\partial x^2} dz\,dx\,dt + \\
&\qquad\qquad - \int_{t_1}^{t_2} \int_{x_1}^{x_2} \int_{-B}^{h} \delta\Phi \frac{\partial \Phi}{\partial x} \delta(z+B) \frac{\partial B}{\partial x} dz\,dx\,dt + \\
&\qquad\qquad\qquad - \int_{t_1}^{t_2} \int_{x_1}^{x_2} \int_{-B}^{h} \delta\Phi \frac{\partial \Phi}{\partial x} \delta(h-z) \frac{\partial h}{\partial x} dz\,dx\,dt \\
&= - \int_{t_1}^{t_2} \int_{x_1}^{x_2} \int_{-B}^{h} \delta\Phi \frac{\partial^2 \Phi}{\partial x^2} dz\,dx\,dt + \\
&\qquad\qquad - \int_{t_1}^{t_2} \int_{x_1}^{x_2} \delta\Phi(-B) \frac{\partial \Phi}{\partial x}(-B) \frac{\partial h}{\partial x}(-B)\,dx\,dt + \\
&\qquad\qquad\qquad + \int_{t_1}^{t_2} \int_{x_1}^{x_2} \delta\Phi(h) \frac{\partial \Phi}{\partial x}(h) \frac{\partial h}{\partial x}(h)\,dx\,dt.
\end{aligned}
$$

The term (B) needs only an integration by parts:

$$
\begin{aligned}
\text{(B)} &\overset{(def.)}{=} \int_{t_1}^{t_2} \int_{x_1}^{x_2} \int_{-B}^{h} \frac{\partial \Phi}{\partial z} \frac{\partial(\delta\Phi)}{\partial z} dz\,dx\,dt \\
&\overset{(int.by\,parts)}{=} \int_{t_1}^{t_2} \int_{x_1}^{x_2} \left[\frac{\partial \Phi}{\partial z} \delta\Phi \Big|_{-B}^{h} - \int_{-B}^{h} \frac{\partial^2 \Phi}{\partial z^2} \delta\Phi\,dz \right] dx\,dt \\
&= \int_{t_1}^{t_2} \int_{x_1}^{x_2} \left[\frac{\partial \Phi}{\partial z}(h)\,\delta\Phi(h) - \frac{\partial \Phi}{\partial z}(-B)\,\delta\Phi(-B) - \int_{-B}^{h} \frac{\partial^2 \Phi}{\partial z^2} \delta\Phi\,dz \right] dx\,dt.
\end{aligned}
$$

Before computing the term (C) we apply the Leibniz rule (B.17) to $\delta\Phi$:

$$
\frac{\partial}{\partial t} \int_{-B(x,t)}^{h(x,t)} \delta\Phi(x,z,t)\,dz = \frac{\partial h}{\partial t}\,\delta\Phi(x,h,t) + \frac{\partial B}{\partial t}\,\delta\Phi(x,-B,t) + \int_{-B}^{h} \frac{\partial\,\delta\Phi}{\partial t}\,dz. \tag{B.18}
$$

By rearranging the term (C), we get:

$$
\begin{aligned}
\text{(C)} &\overset{(def.)}{=} \int_{t_1}^{t_2} \int_{x_1}^{x_2} \int_{-B}^{h} \frac{\partial(\delta\Phi)}{\partial t} dz\,dx\,dt \\
&\overset{(B.18)}{=} \int_{t_1}^{t_2} \int_{x_1}^{x_2} \left[\frac{\partial}{\partial t} \left(\int_{-B}^{h} \delta\Phi\,dz \right) - \frac{\partial h}{\partial t}\,\delta\Phi(h) - \frac{\partial B}{\partial t}\,\delta\Phi(-B) \right] dx\,dt \\
&= \int_{t_1}^{t_2} \int_{x_1}^{x_2} \frac{\partial}{\partial t} \left(\int_{-B}^{h} \delta\Phi\,dz \right) dx\,dt - \int_{t_1}^{t_2} \int_{x_1}^{x_2} \left(\frac{\partial h}{\partial t}\,\delta\Phi(h) + \frac{\partial B}{\partial t}\,\delta\Phi(-B) \right) dx\,dt \\
&= \int_{t_1}^{t_2} \frac{\partial}{\partial t} \left(\int_{x_1}^{x_2} \int_{-B}^{h} \delta\Phi\,dz\,dx \right) dt - \int_{t_1}^{t_2} \int_{x_1}^{x_2} \left(\frac{\partial h}{\partial t}\,\delta\Phi(h) + \frac{\partial B}{\partial t}\,\delta\Phi(-B) \right) dx\,dt
\end{aligned}
$$

$$
\begin{aligned}
&= \int_{x_1}^{x_2}\int_{-B}^{h} \delta\Phi \, dz \, dx \Big|_{t_1}^{t_2} - \int_{t_1}^{t_2}\int_{x_1}^{x_2} \left(\frac{\partial h}{\partial t}\, \delta\Phi(h) + \frac{\partial B}{\partial t}\, \delta\Phi(-B) \right) dx \, dt \\
&= \int_{x_1}^{x_2} \left(\int_{-B(t_2)}^{h(t_2)} \underbrace{\delta\Phi(t_2)}_{=0} \, dz - \int_{-B(t_1)}^{h(t_1)} \underbrace{\delta\Phi(t_1)}_{=0} \, dz \right) dx \\
&\quad - \int_{t_1}^{t_2}\int_{x_1}^{x_2} \left(\frac{\partial h}{\partial t}\, \delta\Phi(h) + \frac{\partial B}{\partial t}\, \delta\Phi(-B) \right) dx \, dt \\
&= - \int_{t_1}^{t_2}\int_{x_1}^{x_2} \left(\frac{\partial h}{\partial t}\, \delta\Phi(h) - \frac{\partial(-B)}{\partial t}\, \delta\Phi(-B) \right) dx \, dt.
\end{aligned}
$$

Since in our applications the function B does not depend on time, the second term above is negligible.

Finally, we gather the three terms (A), (B), and (C) and continue the computation of the variations of $\mathcal{L}$ by rearranging some terms:

$$
\begin{aligned}
&\frac{d}{d\varepsilon}\mathcal{L}(\Phi_\varepsilon, h, B)\Big|_{\varepsilon=0} = \cdots = \text{(A)} + \text{(B)} + \text{(C)} \\
&\quad = \int_{t_1}^{t_2}\int_{x_1}^{x_2} \left[\delta\Phi(h)\, \Phi_x(h)\, h_x(h) - \delta\Phi(-B)\, \Phi_x(-B)\, h_x(-B) - \int_{-B}^{h} \delta\Phi\, \Phi_{xx}\, dz \right] dx\, dt + \\
&\qquad + \int_{t_1}^{t_2}\int_{x_1}^{x_2} \left[\delta\Phi(h)\, \Phi_z(h) - \delta\Phi(-B)\, \Phi_z(-B) - \int_{-B}^{h} \delta\Phi\, \Phi_{zz}\, dz \right] dx\, dt + \\
&\qquad - \int_{t_1}^{t_2}\int_{x_1}^{x_2} \delta\Phi(h)\, h_t \, dx\, dt \\
&\quad \overset{(*)}{=} \int_{t_1}^{t_2}\int_{x_1}^{x_2} \delta\Phi(h) \Big(\Phi_x(h)\, h_x(h) - \Phi_z(h) + h_t \Big) \, dx\, dt + \\
&\qquad + \int_{t_1}^{t_2}\int_{x_1}^{x_2} \delta\Phi(-B) \Big(\Phi_x(-B)\, h_x(-B) + \Phi_z(-B) \Big) \, dx\, dt + \\
&\qquad - \int_{t_1}^{t_2}\int_{x_1}^{x_2}\int_{-B}^{h} \delta\Phi \left(\Phi_{xx} + \Phi_{zz} \right) dz\, dx\, dt,
\end{aligned}
$$

at $(*)$ we gather the terms where the functions are evaluated in h or in $-B$ and the triple integrals. For the Fundamental Lemma of Calculus of Variations, the stationary points of $\mathcal{L}$ with respect to the variations of Φ are those that nullify the three terms multiplied by $\delta\Phi$, therefore, are those that verify the following conditions

$$
\begin{aligned}
\Phi_z &= h_t + \Phi_x\, h_x, \quad && z = h, \\
\Phi_x\, h_x + \Phi_z &= 0, \quad && z = -B, \\
\Phi_{xx} + \Phi_{zz} &= 0, \quad && -B < z < h,
\end{aligned}
$$

which correspond to the kinematic boundary conditions at the free surface (B.11) and at the bottom (B.13) respectively, and to the incompressibility condition of the fluid (B.9). In summary, we showed that the equations for the classical water waves problem are the minimum of the Luke's Lagrangian of Eq.(B.14).

We focus on the last two terms of the Lagrangian density L in Eq.(B.14). We

rearrange the term $\int \Phi_t \, dz$ by using the Leibniz rule reported in Eq. (B.17)

$$\frac{\partial}{\partial t}\int_{-B}^{h} \Phi \, dz = h_t \, \Phi(h) + B_t \, \Phi(-B) + \int_{-B}^{h} \Phi_t \, dz$$
$$\Downarrow$$
$$\int_{-B}^{h} \Phi_t \, dz = \frac{\partial}{\partial t}\int_{-B}^{h} \Phi \, dz - h_t \, \Phi(h) - B_t \, \Phi(-B).$$

We notice that the first term on the right-hand side is negligible, in fact, we observe that

$$\begin{aligned}\int_{t_1}^{t_2}\int_{x_1}^{x_2} \frac{\partial}{\partial t}\left(\int_{-B}^{h} \Phi \, dz\right) dx \, dt &= \int_{x_1}^{x_2}\int_{t_1}^{t_2} \frac{\partial}{\partial t}\left(\int_{-B}^{h} \Phi \, dz\right) dt \, dx \\ &= \int_{x_1}^{x_2} \frac{\partial}{\partial t}\left(\int_{t_1}^{t_2}\int_{-B}^{h} \Phi \, dz \, dt\right) dx \\ &= \int_{x_1}^{x_2} \left(\int_{-B(t_2)}^{h(t_2)} \Phi(t_2) \, dz - \int_{-B(t_1)}^{h(t_1)} \Phi(t_1) \, dz\right) dx\end{aligned}$$

and the terms of the horizontal and temporal boundaries are negligible because they do not contribute to variations of the functional. By using this observation, we approach another expression of L:

$$\begin{aligned}L &= h_t \, \Phi(h) + B_t \, \Phi(-B) - \frac{1}{2} g \, z^2 \Big|_{-B}^{h} - \int_{-B}^{h} \frac{1}{2}\left(\Phi_x^2 + \Phi_z^2\right) dz \\ &= h_t \, \Phi(h) + B_t \, \Phi(-B) - \frac{1}{2} g \, h^2 + \frac{1}{2} g \, B^2 - \int_{-B}^{h} \frac{1}{2}\left(\Phi_x^2 + \Phi_z^2\right) dz.\end{aligned}$$

In order to simplify the notation, we introduce 'tildes' and 'wedges' to denote, respectively, the quantities evaluated at the free surface $z = h$ and at the bottom $z = -B$, for example

$$\tilde{\Phi} := \Phi(h), \qquad \check{\Phi} := \Phi(-B),$$

hence we write the Lagrangian density in a simpler way as follows:

$$L = h_t \tilde{\Phi} + B_t \, \check{\Phi} - \frac{1}{2} g \, h^2 + \frac{1}{2} g \, B^2 - \int_{-B}^{h} \frac{1}{2}\left(\Phi_x^2 + \Phi_z^2\right) dz \tag{B.19}$$

We highlight that the term $\frac{1}{2} g B^2$ might be omitted because, being B prescribed, it does not contribute to the variational process.

Generally, variational formulations involving as few dependent variables as possible are regarded as simpler [279]. It is understandably tempting to solve exactly (i.e. analytically) as many equations as possible in order to 'improve' the solution accuracy, however, this is not always a good idea. In the framework of numerical analysis and scientific computing, there are many examples of efficient and most used algorithms that do exactly the opposite. These are called **relaxation methods** and have proven to be very efficient for stiff problems. When solving numerically a system of equations, the exact solution of a few equations does not necessarily ensure that the overall error

is reduced, and what really matters is that the global error is minimized. A similar idea of relaxation might also be applied to analytical approximations.

In the following, we release the constraints of exact irrotationality and exact incompressibility because approximations of these relations are sufficient in most practical cases, illustrating the advantages of using a variational principle that involves *as many dependent variables as possible*. First, we provide the framework in which the equations for water waves are derived, second, we present two examples of the derivation of the equations for shallow water approximations.

B.4 Relaxed variational principles

The variational formulation of Eq. (B.19) (or, equivalently Eq. (B.14)) imposes that any approximation is exactly irrotational, i.e. the choice of an ansatz (that is, an assumed expression to be verified a posteriori) for Φ necessarily implies an irrotational motion. While keeping an exact formulation, the variational principle is relaxed in order to incorporate explicitly more degrees of freedom, and this modification yields to the Hamilton principle in its most general form. By explicitly introducing the horizontal and vertical velocities $(u,\, v) = (\Phi_x,\, \Phi_z)$, the Lagrangian density may be reformulated as follows:

$$L = h_t\tilde{\Phi} + B_t\,\check{\Phi} - \frac{1}{2}g\,h^2 - \int_{-B}^{h}\left[\frac{1}{2}\Big(u^2+v^2\Big) + \mu(\Phi_x - u) + \nu(\Phi_z - v)\right]dz \tag{B.20}$$

We observe that this Lagrangian density is a first-degree equation in Φ, whereas Eq. (B.14) is a second-degree equation in Φ. The two terms $\mu(\Phi_x - u)$ and $\nu(\Phi_z - v)$ in (B.20) are conditions that force the velocity $\mathbf{u}$ to be the derivative of Φ, whereas μ and ν, called **Lagrange multipliers**, are new variables to determine. By considering variations with respect to u and v one finds the Lagrange multipliers definition. Since

$$\frac{d}{d\varepsilon}\mathcal{L}(\Phi, h, u_\varepsilon, v_\varepsilon, \mu, \nu)\bigg|_{\varepsilon=0} = \int_{t_1}^{t_2}\int_{x_1}^{x_2}\frac{d}{d\varepsilon}L(\Phi, h, u_\varepsilon, v_\varepsilon, \mu, \nu)\bigg|_{\varepsilon=0} dx\,dt$$

we reduce to study the variations of L:

$$\begin{aligned}
\frac{d}{d\varepsilon}L(\Phi,h,u_\varepsilon,v_\varepsilon,\mu,\nu)\bigg|_{\varepsilon=0} &= -\frac{d}{d\varepsilon}\int_{-B}^{h}\left\{\frac{1}{2}\Big[(u+\varepsilon\,\delta u)^2 + (v+\varepsilon\,\delta v)^2\Big] + \right.\\
&\quad \left. + \mu\Big[\Phi_x - (u+\varepsilon\,\delta u)\Big] + \nu\Big[\Phi_z - (v+\varepsilon\,\delta v)\Big]\right\}dz\bigg|_{\varepsilon=0}\\
&= -\frac{d}{d\varepsilon}\int_{-B}^{h}\left\{\frac{1}{2}\Big[\big(u^2 + 2\,\varepsilon\,u\,\delta u + \varepsilon^2(\delta u)^2\big) + \big(v^2 + 2\,\varepsilon\,v\,\delta v + \varepsilon^2(\delta v)^2\big)\Big] + \right.\\
&\quad \left. + \mu\Big[\Phi_x - u - \varepsilon\,\delta u\Big] + \nu\Big[\Phi_z - v - \varepsilon\,\delta v\Big]\right\}dz\bigg|_{\varepsilon=0}\\
&= -\int_{-B}^{h}\left\{\frac{1}{2}\Big[\big(2\,u\,\delta u + 2\,\varepsilon\,(\delta u)^2\big) + \big(2\,v\,\delta v + 2\,\varepsilon\,(\delta v)^2\big)\Big] - \mu\,\delta u - \nu\,\delta v\right\}dz\bigg|_{\varepsilon=0}\\
&= -\int_{-B}^{h}\Big\{\,u\,\delta u + v\,\delta v - \mu\,\delta u - \nu\,\delta v\Big\}dz\\
&= -\int_{-B}^{h}\Big\{\,\delta u\,(u-\mu) + \delta v\,(v-\nu)\Big\}dz;
\end{aligned}$$

from this we conclude that

$$\mu = u \quad \text{and} \quad \nu = v. \tag{B.21}$$

By using these definitions of the Lagrangian multipliers in Eq. (B.20), a new expression of the Lagrangian density descends:

$$L = h_t\tilde{\Phi} + B_t\,\check{\Phi} - \frac{1}{2}g\,h^2 + \int_{-B}^{h}\left[\frac{1}{2}\Big(u^2+v^2\Big) - \mu\,\Phi_x - \nu\,\Phi_z\right]dz \tag{B.22}$$

Anyway, keeping the most general form of the Lagrangian (B.20) is more advantageous because it allows choosing ansatz for Lagrange multipliers different from the previous ones (B.21). Indeed, the Lagrangian (B.20) involves six variables $\{\Phi, h, u, v, \mu, \nu\}$ whereas the simplified Lagrangian (B.22) involves only four variables $\{\Phi, h, u, v\}$ and the original Lagrangian of Eq. (B.14) depends only on two $\{\Phi, h\}$. Extra variables introduce additional freedom in the construction of approximations, thus allowing more subordinate relations to be fulfilled [4]. The more general Lagrange density of Eq. (B.20) provides more flexibility to derive model equations.

The connection between the relaxed Lagrangian of Eq. (B.20) and the variational formulation of the classical mechanics may be seen by applying Green's theorem to Eq. (B.20) (and by reminding the property of equivalence reported in Eq. (B.1)) that yields to another equivalent variational formulation involving the Lagrangian density. We make some computations starting from Eq. (B.20)

$$\begin{aligned}
L &= \underbrace{h_t\,\tilde{\Phi} + B_t\,\check{\Phi} - \frac{1}{2}g\,h^2 - \int_{-B}^{h}\left[\frac{1}{2}\Big(u^2+v^2\Big)}_{\star} + \mu\Phi_x\,\underbrace{-\mu u}_{\star} + \nu\Phi_z\,\underbrace{-\nu v}_{\star}\right]dz \\
&= \star - \int_{-B}^{h}\Big(\mu\,\partial_x\Phi + \nu\,\partial_z\Phi\Big)\,dz \\
&\overset{(i)}{=} \star - \int_{-B}^{h}\Big(\partial_x(\mu\,\Phi)\,\underbrace{-(\partial_x\mu)\Phi}_{\diamondsuit} + \partial_z(\nu\,\Phi)\,\underbrace{-(\partial_z\nu)\Phi}_{\diamondsuit}\Big)\,dz \\
&= \star + \diamondsuit - \int_{-B}^{h}\partial_x(\mu\,\Phi)\,dz - \int_{-B}^{h}\partial_z(\nu\,\Phi)\,dz \\
&\overset{(ii)}{=} \star + \diamondsuit - \Big[\underbrace{\partial_x\int_{-B}^{h}\mu\,\Phi\,dz}_{\text{negligible}} - \partial_x h\,\Big((\mu\,\Phi)(h)\Big) + \partial_x(-B)\,\Big((\mu\,\Phi)(-B)\Big)\Big] - \big(\nu\,\Phi\big)\Big|_{-B}^{h} \\
&= \star + \diamondsuit - \Big[-h_x\tilde{\mu}\,\tilde{\Phi} - B_x\check{\mu}\check{\Phi}\Big] - \tilde{\nu}\,\tilde{\Phi} + \check{\nu}\,\check{\Phi} \\
&\overset{(iii)}{=} \underbrace{h_t\,\tilde{\Phi}}_{\heartsuit} + \underbrace{B_t\,\check{\Phi}}_{\clubsuit} - \frac{1}{2}g\,h^2 - \int_{-B}^{h}\left[\frac{1}{2}\Big(u^2+v^2\Big) - \mu u - \nu v\right]dz \\
&\quad + \int_{-B}^{h}\Big(\partial_x\mu + \partial_z\nu\Big)\Phi\,dz + \underbrace{h_x\tilde{\mu}\,\tilde{\Phi}}_{\heartsuit} + \underbrace{B_x\check{\mu}\check{\Phi}}_{\clubsuit} - \underbrace{\tilde{\nu}\,\tilde{\Phi}}_{\heartsuit} + \underbrace{\check{\nu}\,\check{\Phi}}_{\clubsuit} \\
&= \underbrace{\Big(h_t + h_x\,\tilde{\mu} - \tilde{\nu}\Big)\tilde{\Phi}}_{\heartsuit} + \underbrace{\Big(B_t + B_x\,\check{\mu} + \check{\nu}\Big)\check{\Phi}}_{\clubsuit} - \frac{1}{2}g\,h^2 +
\end{aligned}$$

[4] The Lagrangian density of Eq. (B.22) was used by Kim et al. [149] to derive the 'irrotational' Green-Naghdi equations for long waves in shallow water conditions.

$$+\int_{-B}^{h}\left[\mu\,u-\frac{1}{2}u^2+\nu\,v-\frac{1}{2}v^2+\big(\mu_x+\nu_z\big)\phi\right]dz$$

where at (i) we used the expression for the derivative of the product, at (ii) we used the Leibniz rule (B.17) and at (iii) we have explicited all the terms. From the previous computations, the following Lagrangian density descends:

$$\begin{aligned}L=\big(h_t+\tilde{\mu}h_x-\tilde{\nu}\big)\tilde{\Phi}+\big(B_t+\check{\mu}B_x-\check{\nu}\big)\check{\Phi}-\frac{1}{2}gh^2\\+\int_{-B}^{h}\left[\mu u-\frac{1}{2}u^2+\nu v-\frac{1}{2}v^2+\big(\mu_x+\nu_z\big)\Phi\right]dz.\end{aligned}\tag{B.23}$$

Note that the relaxed variational formulations involving Eq. (B.20) and Eq. (B.23) are strictly equivalent, so one should use the most convenient depending on the problem considered.

B.5 Relaxation in shallow-water regime

The potential of the relaxed variational principles is shown in the following deriving two different models in the shallow-water regime, a classical one and another that allows describing the effects of significant variations in topography.

We call the total depth H and we denote with 'bars' the quantity averaged over the water depth, such as:

$$H(x,t):=h(x,t)+B(x),\qquad \bar{u}=\frac{1}{H}\int_{-B}^{h}u\,dz.$$

For simplicity, in this section, we assume that bathymetry is constant in time.

B.5.1 Choice of a simple ansatz

We begin choosing some approximated, but still physically relevant, representations of all the dependent variables enabling us to avoid the integral in the Lagrangian density expression. To this end, we consider an ansatz of the polynomial type that consists of a zeroth-order polynomial in z both for Φ and u and of a first-order polynomial for v, namely we approximate flows that are almost uniform along the vertical direction. According to that, our ansatz express as follows:

$$\Phi\approx\bar{\Phi}(x,t),\quad u\approx\bar{u}(x,t),\quad v\approx\frac{z+B}{H}\tilde{v}(x,t),\tag{B.24}$$

and these ansatz also mean that we are considering a laminar flow with a linear velocity profile for v. The ansatz of Eq. (B.24) are the basis of most shallow-water approximations. Since for the exact solution $\mu=u$ and $\nu=v$ (we refer to Eq. (B.21)), natural ansatz for the Lagrange multipliers are

$$\mu\approx\bar{\mu},\qquad \nu\approx\frac{z+B}{H}\tilde{\nu}(x,t)\tag{B.25}$$

With the ansatz in Eqs. (B.24–B.25), the Lagrangian density $L(\Phi,h,B,u,v,\mu,\nu)$ of Eq. (B.23) becomes

$$L=\big(h_t+\bar{\mu}h_x+B_t+\bar{\mu}B_x\big)\bar{\Phi}-\frac{1}{2}gh^2+H\left[\bar{\mu}\bar{u}-\frac{1}{2}\bar{u}^2+\frac{1}{3}\tilde{\nu}\tilde{v}-\frac{1}{6}\tilde{v}^2+\bar{\Phi}\bar{\mu}_x\right].\tag{B.26}$$

By applying the formula of the derivative of the product

$$\bar{\Phi} H \bar{\mu}_x + \bar{\Phi} \bar{\mu} H_x = \left(\bar{\Phi} \bar{\mu} H\right) - \bar{\mu} H \bar{\Phi}_x,$$

and by using the property of equivalence of the Lagrangian for the total derivatives, we finally get:

$$L = \bar{\Phi} h_t - \frac{1}{2} g h^2 + H \left[\bar{\mu}\, \bar{u} - \frac{1}{2} \bar{u}^2 + \frac{1}{3} \tilde{\nu} \tilde{v} - \frac{1}{6} \tilde{v}^2 - \bar{\mu} \bar{\Phi}_x \right]. \tag{B.27}$$

The two Lagrangian densities of Eqs. (B.26) and (B.27) lead to the same equations, thus, depending on the constraints, we chose the Lagrangian density that leads to the simplest expression.

B.5.2 Unconstrained approximation

Without imposing any further constraint, we investigate the equations of motion we can obtain from Eq. (B.27) by applying the calculus of variations (we do not enter in the details of these calculations because they are similar to those seen previously and very simple):

$$\begin{aligned}
&\delta \bar{u} : \; 0 = \bar{\mu} - \bar{u} \implies \bar{\mu} = \bar{u}, \\
&\delta \tilde{v} : \; 0 = \tilde{\nu} - \tilde{v} \implies \tilde{\nu} = \tilde{v}, \\
&\delta \bar{\mu} : \; 0 = \bar{u} - \bar{\Phi}_x \implies \bar{u} = \bar{\Phi}_x = \bar{\mu}, \\
&\delta \tilde{\nu} : \; 0 = \tilde{v}, \\
&\delta \bar{\Phi} : \; 0 = h_t + \left(H \bar{\mu}\right)_x, \\
&\delta \bar{h} : \; 0 = \bar{\mu}\, \bar{u} - \frac{1}{2} \bar{u}^2 + \frac{1}{3} \tilde{\nu} \tilde{v} - \frac{1}{6} \tilde{v}^2 - \bar{\mu} \bar{\Phi}_x - \bar{\Phi}_t - g h.
\end{aligned}$$

From the first four relations descends that the motion is irrotational, in fact, the vertical component of velocity is null

$$v \overset{(B.24)}{\approx} \frac{z + B}{H} \tilde{v}(x,t) = 0 = \frac{z + B}{H} \tilde{\nu}(x,t) \overset{(B.25)}{\approx} \nu,$$

and the horizontal component of velocity is constant along the vertical direction

$$u \overset{(B.24)}{\approx} \bar{u}(x,t) = \bar{\mu}(x,t) \overset{(B.25)}{=} \mu,$$

therefore the rotors of the velocity fields are null. Instead, fluid incompressibility is not satisfied identically because, even if $v_z = 0$ and $\nu_z = 0$, the partial derivatives of the horizontal velocity fields, namely u_x and μ_x, may not be negligible. Thanks to this four relations, the last two conditions lead to the classical shallow water equations. In fact, the fifth equation leads to the equation for the mass conservation

$$h_t + \left(H \bar{\mu}\right)_x = (h + B)_t + \left(H \bar{\mu}\right)_x = H_t + \left(H \bar{u}\right)_x = 0$$

because we have a bathymetry constant in time and $\mu = u$. The sixth equation leads to the momentum equation:

$$\bar{\mu}\, \bar{u} - \frac{1}{2} \bar{u}^2 + \frac{1}{3} \tilde{\nu} \tilde{v} - \frac{1}{6} \tilde{v}^2 - \bar{\mu} \bar{\Phi}_x - \bar{\Phi}_t - g h = 0$$

$$
\begin{aligned}
&\cancel{\bar{u}^2} - \frac{1}{2}\bar{u}^2 + \frac{1}{3}\underbrace{\tilde{v}^2}_{=0} - \frac{1}{6}\underbrace{\tilde{v}^2}_{=0} - \cancel{\bar{u}^2} - \bar{\Phi}_t - gh = 0 \\
&\bar{\Phi}_t + \frac{1}{2}\bar{u}^2 + gh = 0 \\
&\partial_x\bar{\Phi}_t + \frac{1}{2}\partial_x\bar{u}^2 + gh_x = 0 \quad \text{(deriv. with respect to } x\text{)} \\
&\partial_t\bar{\Phi}_x + \frac{1}{2}\partial_x\bar{u}^2 + gh_x = 0 \\
&\partial_t\bar{u} + \frac{1}{2}\partial_x\bar{u}^2 + gh_x = 0.
\end{aligned}
$$

B.5.3 Choice of shallow water ansatz

For the last example, we choose a different shallow water ansatz, in which both the velocity field and the velocity potential are independent from the vertical coordinate z, namely such that:

$$\Phi \approx \bar{\Phi}(x,t), \quad u = \mu \approx \bar{u}(x,t), \quad v = \nu \approx \check{v}(x,t) \tag{B.28}$$

where we assumed that the pseudo-velocities are equal to the velocity fields $\mu = u$, $\nu = v$. Physically, the previous conditions correspond to a so-called *columnar flow*, which is a model used for long waves in shallow water regimes as long as their amplitudes are not too large. From a mathematical point of view, the ansatz of Eq. (B.28) imply that the vertical variations of the velocity field do not contribute to the Lagrangian of Eq. (B.23) and hence are negligible. Thus, by assuming the ansatz of Eq. (B.28), the Lagrangian density written in Eq. (B.23) becomes:

$$L = \Big(h_t + \bar{u}h_x + h\bar{u}_x\Big)\bar{\Phi} - \frac{1}{2}gh^2 + \frac{1}{2}h\big(\bar{u}^2 + \check{v}^2\big). \tag{B.29}$$

B.5.4 Constraining with impermeability condition

Since we are considering a columnar flow model, each vertical water column can be considered as a moving rigid body. In presence of bathymetry variations, the columnar flow paradigm then yields that the fluid vertical velocity must be equal to that at the bottom because the bottom is impermeable. Thus, we require that the fluid particles follow the bottom profile, i.e.

$$\check{v} = -B_t - \bar{u}B_x,$$

this identity being the bottom impermeability condition (see Eq. (B.13)) expressed with the ansatz of Eq. (B.28).

After substituting the previous relation into the Lagrangian density of Eq. (B.29) we obtain the following set of equations:

$$
\begin{aligned}
&\delta\bar{u} : \ 0 = \bar{u} - \bar{\Phi}_x - \check{v}B_x, \\
&\delta\bar{\Phi} : \ 0 = h_t + (h\bar{u})_x, \\
&\delta h : \ 0 = \bar{\Phi}_t + gh + \bar{u}\bar{\Phi}_x - \frac{1}{2}\Big(\bar{u}^2 + \check{v}^2\Big).
\end{aligned}
$$

By taking the gradient of the last condition and by eliminating $\bar{\Phi}$ from the first equation, we obtain the system of governing equations:

$$\begin{cases} h_t + \left(h\bar{u}\right)_x = 0 \\ \left(\bar{u} - \check{v}B_x\right)_t + \left(\frac{1}{2}\bar{u}^2 + \frac{1}{2}\check{v}^2 + gh\right)_x = 0 \end{cases}$$

together with the auxiliary relations

$$\bar{u} = \bar{\Phi}_x + \check{v}B_x, \quad \check{v} = -B_t + \bar{u}B_x.$$

The surface waves moving on an irregular or steep bottom have always attracted the particular attention of researchers. One of the first studies in this direction is due to Dressler [74] that created a model which included the curvature effects, requiring the computation of the second-order derivatives of the bottom's profile; for irregular shapes of the bottom, this becomes problematic. The previous approximation proposed in §B.5.3 and §B.5.4 tries to improve the classical equations by including a better representation of the bottom shape.

References

[1] G. Alessio, B. Behncke, E. Biagioli, S. Branca, M. L. Carapezza, M. Cerminara, G. Currenti, G. D'Addezio, M. De Lucia, M. de' Michieli Vitturi, M. A. Di Vito, A. Gasparini, E. Marotta, R. Nappi, T. Esposti Ongaro, M. Locritani, F. Muccini, M. Ranaldi, G. P. Ricciardi, F. Sansivero, and P. Stefanelli. L'immagine del vulcano; dalle illustrazioni storiche alla grafica computazionale. (The volcano image; from the historical pictures to computational graphics), 2017.

[2] S. Almeland. Implementation of an air-entrainment model in interFoam. *Proceedings of CFD with OpenSource Software, 2018*, 2018. doi: 10.17196/OS_CFD#YEAR_2018.

[3] W. K. Anderson, J. C. Newman, D. L. Whitfield, and E. J. Nielsen. Sensitivity analysis for Navier-Stokes equations on unstructured meshes using complex variables. *AIAA journal*, 39(1):56–63, 2001.

[4] E. Audusse, F. Bouchut, M. O. Bristeau, R. Klein, and B. Perthame. A fast and stable well-balanced scheme with hydrostatic reconstruction for shallow water flows. *SIAM J. Sci. Comput.*, 25:2050–2065, 2004.

[5] M. V. Avolio, G. Mirocle Crisci, S. Di Gregorio, R. Rongo, W. Spataro, and G. A. Trunfio. SCIARA γ2: An improved cellular automata model for lava flows and applications to the 2002 Etnean crisis. *Computers & Geosciences*, 32(7):876–889, aug 2006. doi: 10.1016/j.cageo.2005.10.026.

[6] N. J. Balmforth, R. V. Craster, P. Perona, A. C. Rust, and R. Sassi. Viscoplastic dam breaks and the bostwick consistometer. *Journal of non-Newtonian fluid mechanics*, 142: 63–78, 2007.

[7] D. Barca, G. M. Crisci, S. Di Gregorio, and F. Nicoletta. Cellular automata for simulating lava flows: A method and examples of the Etnean eruptions. *Transport Theory and Statistical Physics*, 23(1-3):195–232, jan 1994. doi: 10.1080/00411459408203862.

[8] D. Barca, G. M. Crisci, R. Rongo, S. Di Gregorio, and W. Spataro. Application of the cellular automata model SCIARA to the 2001 Mount Etna crisis. *American Geophysical Union, Geophysical Monograph Series*, 2004.

[9] G. K. Batchelor. *An introduction to fluid dynamics*. Cambridge University Press, 2000.

[10] L. Begnudelli and B. F. Sanders. Conservative wetting and drying methodology for quadrilateral grid finite-volume models. *Journal of Hydraulic Engineering*, 133(3):312–322, 2007.

[11] J. Behrens. Atmospheric and ocean modeling with an adaptive finite element solver for the shallow-water equations. *Applied Numerical Mathematics*, 26(1-2):217–226, 1998.

[12] N. Bernabeu, P. Saramito, and C. Smutek. Numerical modeling of non-Newtonian viscoplastic flows: part ii. Viscoplastic fluids and general tridimensional topographies. *International Journal of Numerical Analysis & Modeling*, 11:214–229, 2013.

[13] N. Bernabeu, P. Saramito, and C. Smutek. Modelling lava flow advance using a shallow-depth approximation for three-dimensional cooling of viscoplastic flows. *Geological Society, London, Special Publications*, 426(1):409–423, 2016. doi: 10.1144/SP426.27.

[14] S. Bi, J. Zhou, Y. Liu, and L. Song. A finite volume method for modeling shallow flows with wet-dry fronts on adaptive cartesian grids. *Mathematical Problems in Engineering*, 2014, 2014.

[15] E. Biagioli, M. de' Michieli Vitturi, and F. D. Benedetto. Modified shallow water model for viscous fluids and positivity preserving numerical approximation. *Applied Mathematical Modeling*, 94:482–505, 2021. doi: 10.1016/j.apm.2020.12.036.

[16] E. Biagioli, M. de' Michieli Vitturi, and F. D. Benedetto. Benchmarking modified shallow water model for lava flows. 2021.

[17] G. Bilotta, A. Cappello, A. Hérault, A. Vicari, G. Russo, and C. D. Negro. Sensitivity analysis of the MAGFLOW cellular automaton model for lava flow simulation. *Environmental Modelling & Software*, 35:122–131, jul 2012. doi: 10.1016/j.envsoft.2012.02.015.

[18] G. Bilotta, A. Herault, A. Cappello, G. Ganci, and C. D. Negro. SPHLOW: SPH model for lava flows. *Geological Society, London, Special Publications*, 426, 2015.

[19] R. J. Blong. *Volcanic hazards : a sourcebook on the effects of eruptions*. Sydney : Academic Press, 1984.

[20] L. A. Bolshov, V. V. Chudanov, A. G. Popkov, V. F. Strizhov, and P. N. Vabishchevich. Numerical models of molten core spreading processes in nuclear reactor safety problems. *Nuclear Science Journal*, 32:171–179, 1995.

[21] K. Bonne, M. Kervyn, L. Cascone, S. Njome, E. Van Ranst, E. Shu, S. Ayonghe, P. Jacobs, and G. Ernst. A new approach to assess long-term lava flow hazard and risk using GIS and low-cost remote sensing: the case of Mount Cameroon, West Africa. *International Journal of Remote Sensing*, 29:6539–6564, 2008. doi: 10.1080/01431160802167873.

[22] J. P. Boris and D. L. Book. Flux-corrected transport. I. SHASTA, A fluid transport algorithm that works. *Journal of Computational Physics*, 11:38–69, 1973. doi: 10.1016/0021-9991(73)90147-2.

[23] F. Bouchut. *Nonlinear stability of finite Volume Methods for hyperbolic conservation laws: And Well-Balanced schemes for sources*. Springer Science & Business Media, 2004.

[24] J. Boussinesq. *Essai sur la théorie des eaux courantes*. Impr. nationale, 1877.

[25] M. Boutounet, L. Chupin, P. Noble, and J.-P. Vila. Shallow water equations for Newtonian fluids over arbitrary topographies. *Comm. Math. Sci.*, 6(1):29–55, 2008.

[26] M. E. Braaten. *Development and evaluation of iterative and direct methods for the solution of the equations governing recirculating flows*. PhD thesis, University of Minnesota, 1985.

[27] J. U. Brackbill, D. B. Kothe, and C. Zemach. A continuum method for modeling surface tension. *Journal of computational physics*, 100:335–354, 1992.

[28] S. F. Bradford and B. F. Sanders. Finite-volume model for shallow-water flooding of arbitrary topography. *Journal of Hydraulic Engineering*, 128(3):289–298, 2002.

[29] D. Bresch and P. Noble. Mathematical justification of a shallow water model. *Methods Appl. Anal.*, 14(2):87–118, 2007.

[30] A. Cappello, G. Ganci, S. Calvari, N. M. Pérez, P. A. Hernández, S. V. Silva, and J. C. D. N. Cabral. Lava flow hazard modeling during the 2014–2015 Fogo eruption, Cape Verde. *J. Geophys. Res. Solid Earth*, 121:2290–2303, 2016. doi: 10.1002/2015JB012666.

[31] A. Carati and L. Galgani. *Appunti di Meccanica Analitica*. 2013–2014. URL http://www.mat.unimi.it/users/carati/didattica/dispense.

[32] L. S. Caretto, R. M. Curr, and D. B. Spalding. Two numerical methods for three-dimensional boundary layers. *Computer Methods in Applied Mechanics and Engineering*, 1:39–57, 1972.

[33] L. S. Caretto, A. D. Gosman, S. V. Patankar, and D. B. Spalding. Two calculation procedures for steady, three-dimensional flows with recirculation. *Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics. Lecture Notes in Physics*, 19, 1973. doi: 10.1007/BFb0112677.

[34] V. P. Carey and J. C. Mollendorf. Measured variation of thermal boundary-layer thickness with Prandtl number for laminar natural convection from a vertical uniform-heat-flux surface. *Inc. J. Heat Mass Tnmfer.*, 21:481–488, 1978.

[35] M. J. Castra Día, E. D. Fernández-Nieto, and A. Ferreiro. Sediment transport models in hallow water equations and numerical approach by high order finite volume methods. *Computers & Fluids*, 37(3):299–316, 2008. doi: 10.1016/j.compfluid.2007.07.017.

[36] A. T. Chatfield and C. G. Reddick. Understanding Risk Communication Gaps through E-Government Website and Twitter Hashtag Content Analyses: The Case of Indonesia's Mt. Sinabung Eruption. *Homeland Security & Emergency Management*, 12, 2015. doi: 10.1515/jhsem-2014-0086.

[37] D. K. Chester, J. E. Guest, and C. R. J. Kilburn. The rheological behavior of basaltic lavas. *Mount Etna*, pages 187–228, 1985.

[38] D. K. Chester, A. M. Duncan, P. Wetton, and R. Wetton. Responses of the Anglo-American military authorities to the eruption of Vesuvius, march 1944. *Journal of Historical Geography*, 33:168–196, 2007.

[39] M. O. Chevrel, M. Favalli, N. Villeneuve, A. J. L. Harris, A. Fornaciai, N. Richter, A. Derrien, P. , A. Di Muro, and A. Peltier. Lava flow hazard map of Piton de la Fournaise volcano. *Natural Hazard and Earth System Sciences*, 2020.

[40] A. Chinnayya, A. LeRoux, and N. Seguin. A well-balanced numerical scheme for the approximation of the shallow-water equations with topography: The resonance phenomenon. *Int. J. Finite Volumes.*, 2004.

[41] A. J. Chorin. Solution of implicitly discretized fluid flow equations by operator-splitting. *Math. Comp*, 22:745–762, 1968. doi: 10.1090/S0025-5718-1968-0242392-2.

[42] D. M. Christopher and B. Wang. Prandtl number effects for Marangoni convection over a flat surface. *Int. J. Therm. Sci.*, 40:564–570, 2001. doi: 10.1016/0021-9991(73)90147-2.

[43] C. Cimarelli, A. Costa, S. Mueller, and H. Mader. Rheology of magmas with bimodal crystal size and shape distributions: insights from analogue experiments. *Geochemistry, Geophysics, Geosystems*, 12, 2011. doi: 10.1029/2011GC003606.

[44] C. Cimarelli, A. Costa, S. Mueller, and H. M. Mader. Rheology of magmas with bimodal crystal size and shape distributions: Insights from analog experiments. *Geochemistry, Geophysics, Geosystems*, 12(7), 2011. doi: 10.1029/2011GC003606.

[45] D. Clamond and D. Dutykh. Practical use of variational principles for modeling water waves. *Physica D*, 241:25–36, 2012.

[46] D. Clamond and D. Dutykh. Modified shallow water equations for significantly varying seabeds. *Appl. Math. Mod.*, 40:9767–9787, 2016.

[47] R. Colombrita. Methodology for the construction of earth barriers to divert lava flows: the Mt Etna 1983 eruption. *Bulletin of Volcanology*, 47:1009–1038, 1984.

[48] M. Coltelli, C. Proietti, S. Branca, M. Marsella, D. Andronico, and L. Lodato. Analysis of the 2001 lava flow eruption of Mt. Etna from three-dimensional mapping. *Journal of Geophysical Research*, 112(F2), jun 2007. doi: 10.1029/2006JF000598.

[49] L. J. Connor, C. B. Connor, K. Meliksetian, and I. Savov. Probabilistic approach to modeling lava flow inundation: a lava flow hazard assessment for a nuclear facility in Armenia. *J. Appl. Volcanol.*, 1(1):3, 2012. doi: 10.1186/2191-5040-1-3.

[50] B. Cordonnier, K. U. Hess, K. U. Lavallee, and D. B. Dingwell. Rheological properties of dome lavas: case study of Unzen volcano. *Earth and Planetary Science Letters*, 279: 263–272, 2009.

[51] B. Cordonnier, L. Caricchi, M. Pistone, J. Castro, K. Hess, S. Gottschaller, M. Manga, D. B. Dingwell, and L. Burlini. The viscous-brittle transition of crystal-bearing silicic melt: Direct observation of magma rupture and healing. *Geology*, 40(7):611–614, jun 2012. doi: 10.1130/G3914.1.

[52] B. Cordonnier, E. Lev, and F. Garel. Benchmarking lava-flow models. *Geological Society, London, Special Publications*, 426:425–445, 2015.

[53] A. Costa and G. Macedonio. Nonlinear phenomena in fluids with temperature-dependent viscosity: An hysteresis model for magma flow in conduits. *Geophys Res Lett*, 29(10): 401–404, May 2002. doi: 10.1029/2001GL014493.

[54] A. Costa and G. Macedonio. Viscous heating in fluids with temperature-dependent viscosity: implications for magma flows. *Nonlinear Processes in Geophysics*, 10(6):545–555, 2003. doi: 10.5194/npg-10-545-2003.

[55] A. Costa and G. Macedonio. Numerical simulation of lava flows based on depth-averaged equations. *Geophysical Research Letters*, 32(5), 2005. doi: 10.1029/2004gl021817.

[56] R. Costantinescu, K. G. Zuccoloto, D. Ferrés, C. Siebe, L. Capra, and R. Tonini. Probabilistic multi-hazard assessment at an active but under-monitored volcano: Ceboruco, Mexico. 2021. doi: 10.21203/rs.3.rs-242445/v1.

[57] R. Courant, K. O. Friedrichs, and H. Lewy. Über die partiellen differenzengleichungen der mathematischen physik. *Math. Ann.*, 100:32–74, 1928.

[58] R. Courant, K. O. Friedrichs, and H. Lewy. On the partial difference equations of mathematical physics. *IBM J.*, 11:215—-234, 1967.

[59] R. C. Courtney and R. S. White. Anomalous heat flow and geoid across the Cape Verde Rise: Evidence for dynamic support from a thermal plume in the mantle. *Geophys. J. R. Astron. Soc.*, 87:815–867, 1986.

[60] G. M. Crisci, S. Di Gregorio, O. Pindaro, and G. A. Ranieri. Lava flow simulation by a discrete cellular model: first implementation. *International Journal of Modelling & Simulation*, 6:137–140, 1986.

[61] G. M. Crisci, S. D. Gregorio, F. P. Nicoletta, R. Rongo, and W. Spataro. Cellular automata approaches for simulating rheology of complex geological phenomena. *Cellular Automata: Research Towards Industry*, 1998.

[62] G. M. Crisci, G. Iovine, S. Di Gregorio, and V. Lupiano. Lava-flow hazard on the SE flank of Mt. Etna (Southern Italy). *Journal of Volcanology and Geothermal Research*, 177(4): 778–796, nov 2008. doi: 10.1016/j.jvolgeores.2008.01.041.

[63] J. Crisp and S. Baloga. A model for lava flows with two thermal components. *J. Geophys. Res.*, 95:1255–1270, 1990.

[64] M. L. Damiani, G. Groppelli, G. Norini, E. Bertino, A. Gigliuto, and S. Nucita. A lava flow simulation model for the development of volcanic hazard maps for mount etna (italy). *Computers & Geosciences*, 32:512–526, 2006. doi: 10.1016/j.cageo.2005.08.011.

[65] S. J. Day, S. I. N. Heleno da Silva, and J. F. B. D. Fonseca. A past giant lateral collapse and present-day flank instability of Fogo, Cape Verde Islands. *J. Volcanol. Geotherm. Res.*, 94:191–218, 1999.

[66] M. de' Michieli Vitturi and S. Tarquini. MrLavaLoba: A new probabilistic model for the simulation of lava flows as a settling process. *EGU General Assembly Conference Abstracts*, 21, 2021.

[67] M. de' Michieli Vitturi, T. Esposti Ongaro, G. Lari, and A. Aravena. IMEX_SfloW2D 1.0. a depth-averaged numerical flow model for pyroclastic avalanches. *Geosci. Model Dev.*, 12: 581–595, 2019. doi: 10.5194/gmd-12-581-2019.

[68] C. Del Negro, L. Fortuna, A. Herault, and A. Vicari. Simulations of the 2004 lava flow at Etna volcano using the MAGFLOW cellular automata model. *Bulletin of Volcanology*, 70 (7):805–812, oct 2008. doi: 10.1007/s00445-007-0168-8.

[69] S. S. Deshpande, L. Anumolu, and M. F. Trujillo. Evaluating the performance of the two-phase flow solver interFoam. *Computational Science & Discovery*, 5, 2012.

[70] H. R. Dietterich, E. Lev, J. Chen, J. A. Richardson, and K. V. Cashman. Benchmarking computational fluid dynamics models of lava flow simulation for hazard assessment, forecasting, and risk management. *Journal of Applied Volcanology*, 6(1):9, 2017. doi: 10.1186/s13617-017-0061-x.

[71] S. M. Dionis, N. M. Pérez, P. A. Hernández, G. Melián, F. Rodríguez, E. Padrón, H. Sumino, J. Barrrancos, G. D. Padilla, P. Fernandes, Z. Bandomo, S. V. Silva, J. M. Pereira, H. Semedo, and J. Cabral. Diffuse CO_2 degassing and volcanic activity at Cape Verde islands, West Africa. *Earth Planets Space*, 67, 2015. doi: 10.1186/s40623-015-0219-x.

[72] F. Dobran. Nonequilibrium flow in volcanic conduits and application to the eruptions of mt. st. helens on may 18, 1980, and vesuvius in ad 79. *Journal on Volcanology and Geothermal Research*, 49:285–311, 1992.

[73] M. Dragoni, I. Borsari, and A. Tallarico. A model for the shape of lava flow fronts. *Journal of Geophysical Research: Solid Earth*, 110(B9), 2005.

[74] R. F. Dressler. New nonlinear shallow-water equations with curvature. *J. Hydraul. Res.*, 16:205–222, 1978.

[75] H. El Hadri, M. Goujon, and R. Paris. A database of the economic impacts of historical volcanic eruptions. *Études et Documents, CERDI*, 14, 2021.

[76] E. Elgar. *Governing Disasters: The Challenges of Emergency Risk Regulation.*, 2011.

[77] B. Eltard Larsen, D. R. Fuhrman, and J. Roenby. Performance of interFoam on the simulation of progressive waves. *Coastal Engineering Journal*, 61(3):380–400, 2019. doi: 10.1080/21664250.2019.1609713.

[78] B. Eltard Larsen, D. R. Fuhrman, and J. Roenby. Performance of interfoam on the simulation of progressive waves. *Coastal Engineering Journal*, 91(3):380–400, 2019. doi: 10.1080/21664250.2019.1609713.

[79] ENCIT, editor. *On the performance of coupled and segregated methods for solving two-dimensional incompressible flows employing unstructured grids*, 2014.

[80] T. Esposti Ongaro, M. de' Michieli Vitturi, M. Cerminara, B. Calusi, E. Biagioli, and A. Fornaciai. Applicabilità dei modelli numerici per la simulazione di tsunami generati da frane e valanghe piroclastiche a Stromboli (Applicability of numerical models for the simulation of tsunami generated from landslides and pyroclastic avalanches at Stromboli), 2021.

[81] S. A. Fagents, T. K. P. Gregg, and R. M. C. Lopes, editors. *Modeling Volcanic Processes. The Physics and Mathematics of Volcanism.* Cambridge University Press, 2012.

[82] M. T. Farmer, J. J. Sienicki, and B. W. Spencer. The meltspread-1 computer code for the analysis of transient spreading in containments. *Transactions of the American Nuclear Society*, 62:644–646, 1990.

[83] M. Favalli, M. Pareschi, A. Neri, and I. Isola. Forecasting lava flow paths by a stochastic approach. *Geophysical Research Letters*, 35, 2005.

[84] A. Felpeto, J. Marti, and R. Ortiz. Automatic GIS-based system for volcanic hazard assessment. *J Volcanol Geotherm Res*, 166:106–116, 2007. doi: 10.1016/j.jvolgeores.2007.07.008.

[85] E. D. Fernández-Nieto, F. Bouchut, D. Bresch, M. J. C. Diaz, and A. Mangeney. A new Savage-Hutter type model for submarine avalanches and generated tsunami. *Journal of Computational Physics*, 227(16):7720–7754, 2008.

[86] E. D. Fernández-Nieto, P. Noble, and J.-P. Vila. Shallow Water equations for Non-Newtonian fluids. *Journal of Non-Newtonian Fluid Mechanics*, 165:712–732, 2010. doi: 10.1016/j.jnnfm.2010.03.008.

[87] E. D. Fernández-Nieto, P. Noble, and J.-P. Vila. Shallow water equations for power law and Bingham fluids. *Sci. China Math.*, 55:277–283, 2012. doi: 10.1007/s11425-011-4358-7.

[88] S. Ferrari and F. Saleri. A new two-dimensional shallow water model including pressure effects and slow varying bottom topography. *M2AN Math. Model. Numer. Anal.*, 38:211–234, 2004.

[89] J. H. Ferziger and M. Peric. *Computational Methods for Fluid Dynamics*. Springer, 3 edition, 2002.

[90] L. flows and domes: Springer, Berlin Heidelberg, IAVCEI Proceedings in Volcanology, editors. *Viscoplastic models of lava domes*, volume 2, 1990.

[91] M. M. Francois, S. J. Cummins, E. D. Dendy, D. B. Kothe, J. M. Sicilian, and M. W. Williams. A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework. *J. Comp. Phys.*, 213(1):141–173, 2006.

[92] R. Franzke, S. Sebben, T. Bark, E. Willeson, and A. Broniewicz. Evaluation of the Multiple Reference Frame Approach for the Modelling of an Axial Cooling Fan. *MDPI*, 2019.

[93] F. G. Friedlander. *Introduction to the theory of distributions*. Cambridge University Press, 1982.

[94] E. Fujita and M. Nagai. LavaSIM: its physical base and applicability. *Geological Society, London, Special Publications*, 426, 2015. doi: 10.1144/SP426.14.

[95] T. Gallouet, J. M. Hérard, and N. Seguin. Some approximate Godunov schemes to compute shallow-water equations with topography. *Comput. & Fluids*, 32:479–513, 2003.

[96] P. F. Galpin and G. D. Raithby. Numerical solution of problems in incompressible fluid flow: treatment of the temperature-velocity coupling. *Numerical Heat Transfer*, 10:105–129, 1986. doi: 10.1080/10407788608913511.

[97] C. Galusinski and P. Vigneaux. On stability condition for bifluid flows with surface tension: Application to microfluidics. *J. Comp. Phys.*, 227(12):6140–6164, 2008.

[98] G. Ganci, A. Vicari, A. Cappello, and C. D. Negro. An emergent strategy for volcano hazard assessment: From thermal satellite monitoring to lava flow modeling. *Remote Sensing of Environment*, 119:197–207, apr 2012. doi: 10.1016/j.rse.2011.12.021.

[99] F. Garel, E. Kaminski, S. Tait, and A. Limare. An experimental study of the surface thermal signature of hot subaerial isoviscous gravity currents: Implications for thermal monitoring of lava flows and domes. *Journal of Geophysical Research: Solid Earth*, 117, 2012.

[100] I. M. Gelfand and S. V. Fomin. *Calculus of variations*. 1963.

[101] J. Gerbeau and B. Perthame. Derivation of viscous Saint-Venant system for laminar shallow water: Numerical validation. *Discrete and continuous dynamical systems–series B*, 1(1): 89–102, 02 2001.

[102] D. Giordano, J. K. Russell, and D. B. Dingwell. Viscosity of magmatic liquids: A model. *Earth and Planetary Science Letters*, 271:123–134, 2008. doi: 10.1016/j.epsl.2008.03.038.

[103] L. S. Glaze and S. M. Baloga. Simulation of inflated Pahoehoe lava flows. *Journal of Volcanology and Geothermal Research*, 255:108–123, 2013.

[104] S. K. Godunov. *Different Methods for Shock Waves*. PhD thesis, Moscow State University, 1954.

[105] S. K. Godunov. A difference scheme for numerical solution of discontinuous solution of hydrodynamic equations. *Mat. Sbornik*, 47:271–306, 1959.

[106] G. H. Golub and C. F. V. Loan. *Matrix Computations, 3rd ed.* Baltimore, MD: Johns Hopkins, 1996.

[107] H. Gonnermann and M. Manga. The fluid mechanics inside a volcano. *Annual Review of Fluid Mechanics*, 39:321–356, 2007.

[108] S. Gottlieb, C. W. Shu, and E. Tadmor. Strong stability-preserving high order time discretization methods. *SIAM Review*, 43:89–112, 2001.

[109] R. Greenkorn. *Momentum, Heat, and Mass Transfer Fundamentals.* CRC Press, 1 edition, 2018.

[110] R. H. Griffiths. The dynamic of lava flows: annual review of fluid mechanics. *Natural Hazards*, 32:477–518, 2000.

[111] D. Guha-Sapir, R. Below, and P. Hoyois. The CRED/OFDA international disaster database. 2009. URL www.emdat.be.

[112] C. V. Hamilton, L. S. Glaze, M. R. James, and S. M. Baloga. Topographic and stochastic influences on pahoehoe lava lobe emplacement. *Bulletin of Volcanology*, 75:1–16, 2013. doi: 10.1029/2009GL039717.

[113] R. F. Hanby and D. J. Silvester. *Segregated and Coupled Iterative Solution Algorithms for Incompressible Swirling Flow.* PhD thesis, UMIST, 1995.

[114] R. F. Hanby, D. J. Silvester, and J. W. Chew. A comparison of coupled and segregated iterative solution technique for incompressible swirling flow. *Int. J. Numer. Meth. Fluids*, 22:353–373, 1996. doi: 10.1002/(SICI)1097-0363(19960315)22:5<353::AID-FLD327>3.0.CO;2-Z.

[115] F. H. Harlow and J. E. Welsh. Numerical calculation of time dependent viscous incompressible flow with free surface. *Phys. Fluids*, 8:2182–2189, 1965.

[116] A. J. L. Harris and S. Rowland. Flowgo: a kinematic thermo-rheological model for lava flowing in a channel. *Bulletin of Volcanology*, 63:20–44, 2001.

[117] A. J. L. Harris and S. K. Rowland. Effusion rate controls on lava flow length and the role of heat loss: a review. *Studies in Volcanology: The Legacy of George Walker. Special Publications of IAVCEI*, 2:33–51, 2009. doi: 10.1186/s13617-020-00096-.

[118] A. J. L. Harris, M. Favalli, R. Wright, and H. Garbeil. Hazard assessment at mount etna using a hybrid lava flow inundation model and satellite-based land classification. *Natural Hazards*, 58:1001–1027, 2011.

[119] A. J. L. Harris, M. Rhéty, L. Gurioli, N. Villeneuve, and R. Paris. Simulating the thermo-rheological evolution of channel-contained lava: FLOWGO and its implementation in EXCEL. *Geological Society, London, Special Publications*, 426(1):313–336, 2015. ISSN 0305-8719. doi: 10.1144/SP426.9.

[120] A. J. L. Harris, T. De Groeve, F. Garel, and S. A. Carn. Detecting, modelling and responding to effusive eruptions. *Geological Society, London, Special Publications*, 426: 1001–1027, 2016. doi: 10.1144/SP426.29.

[121] A. Herault, A. Vicari, A. Ciraudo, and C. D. Negro. Forecasting lava flow hazards during the 2006 Etna eruption: Using the MAGFLOW cellular automata model. *Computers & Geosciences*, 35(5):1050–1060, may 2009. doi: 10.1016/j.cageo.2007.10.008.

[122] A. Herault, G. Bilotta, A. Vicari, E. Rustico, and C. D. Negro. Numerical simulation of lava flow using a GPU SPH model. *The Lava Flow Invasion Hazard Map at Mount Etna and Methods for its Dynamic Update. Annals of Geophysics*, 54, 2011. doi: 10.4401/ag-5343.

[123] S. J. R. Hergarten. Modelling rapid mass movements using the shallow water equations in Cartesian coordinates. *Nat. Hazards Earth Syst. Sci.*, 15:671–685, 2015. doi: 10.5194/nhess-15-671-2015.

[124] M. Hidaka, N. Sato, and H. Ujita. Verification for flow analysis capability in the model of three-dimensional natural convection with simultaneous spreading, melting and solidification for the debris coolability analysis module in the severe accident analysis code 'SAMPSON'. *Journal of Nuclear Science and Technology*, 39:520–530, 2002. doi: 10.1080/18811248.2002.9715230.

[125] M. Hidaka, A. Goto, S. Umino, and E. Fujita. VTFS project: development of the lava flow simulation code LavaSIM with a model for three-dimensional convection, spreading, and solidification. *Geochemistry, Geophysics, Geosystems*, 6, 2005. doi: 10.1029/10.1029/2004GC000869.

[126] T. Hino. Computation of viscous flows with free surface around an advancing ship. In *Proc. 2nd Osaka Int. Colloquium on Viscous Fluid Dynamics In Ship and Ocean Technology*. Osaka Univ., 1992.

[127] C. W. Hirt and B. D. Nicholls. Volume of fluid (VOF) method for dynamics of free boundaries. *J . Comput. Phys.*, 39:201–221, 1981.

[128] A. J. Hogg and D. Pritchard. The effects of hydraulic resistance on dam-break and other shallow inertial flows. *Journal of Fluid Mechanics*, 501:179–212, 2004.

[129] T. Holzmann. Mathematics, Numerics, Derivations and OpenFOAM®, 2019. URL https://holzmann-cfd.de.

[130] G. Hulme. Generation of magma at lunar impact crater sites. *Nature*, 252:556–558, 1974.

[131] H. E. Huppert. The propagation of two-dimensional and axisymmetric viscous gravity currents over a rigid horizontal surface. *Journal of Fluid Mechanics*, 121:43–58, 1982.

[132] N. Huybrechts, C. Villaret, and J. M. Hervouet. Comparison between 2D and 3D modelling of sediment transport: application to the dune evolution. In *River Flow 2010*, volume 4, pages 887–893, 2010.

[133] S. Hysing. A new implicit surface tension implementation for interfacial flows. *Int.l J. Numer. Meth. Fluids*, 51(6):659–672, 2006.

[134] R. I. Issa. Solution of implicitly discretized fluid flow equations by operator-splitting. *J. Comput. Phys.*, 62:40–65, 1986.

[135] H. Jasak. *Error Analysis and Estimation for the Finite Volume Method with Applications to Fluid Flows*. PhD thesis, Imperial College, University of London, 1996.

[136] H. Jasak, H. G. Weller, and A. D. Gosman. High resolution NVD differencing scheme for arbitrarily unstructured meshes. *International Journal for Numerical Methods in Fluids*, (31):431–449, 1999.

[137] H. Jasak, A. Jemcov, and Z. Tukovic. OpenFOAM: a c++ library for complex physics simulations. In *International Workshop on Coupled Methods in Numerical Dynamics, Dubrovnik, Croatia*, 2007.

[138] S. Jin. A steady-state capturing method for hyperbolic system with geometrical source terms. *Math. Model. Numer. Anal.*, 35:631–645, 2001.

[139] S. Jin and X. Wen. Two interface-type numerical methods for computing hyperbolic systems with geometrical source terms having concentrations. *SIAM J. Sci. Comput.*, 26: 2079–2101, 2005.

[140] K. C. Karki and H. C. Mongia. Evaluation of a coupled solution approach for fluid flow calculations in body fitted co-ordinates. *International Journal for Numerical Methods in Fluids*, 11:1–20, 1990.

[141] J. Kauahikaua, K. Cashman, T. Mattox, C. Heliker, K. Hon, K. Mangan, and C. Thornber. Observation on basaltic lava streams in tubes from Kilauea Volcano, island of Hawai'i. *J. Geophys. Res.*, 103:27,303–27,324, 1998.

[142] J. L. Kavanagh, S. L. Engwell, and S. A. Martin. A review of laboratory and numerical modelling in volcanology. *Solid Earth*, 9:531–571, 2018. doi: 10.5194/se-9-531-2018.

[143] T. Kawamura and H. Miyata. Simulation of nonlinear ship flows by density-function method. *J . Soc. Naval Architects Japan*, 176:1–10, 1994.

[144] K. Kelfoun and T. Druitt. Numerical modeling of the emplacement of socompa rock avalanche, chile. *Journal of Geophysical Research: Solid Earth*, 110:B12202, 2005. doi: 0.1029/2005JB003758.

[145] K. Kelfoun and S. V. Vargas. VolcFlow capabilities and potential development for the simulation of lava flows. *Geological Society, London, Special Publications*, 426(1):337–343, 2016.

[146] S. Keough. Optimising the parallelisation of openfoam simulations. 2014.

[147] L. Keszthelyi. Measurements of the cooling at the base of pahoehoe flows. *Geophys. Res. Lett.*, 22:2195–2198, 1995.

[148] L. Keszthelyi and S. Self. Some physical requirements for the emplacement of long basaltic lava flows. *Journal of Geophysical Research: Solid Earth*, 103(B11):27447–27464, nov 1998. doi: 10.1029/98JB00606.

[149] J. W. Kim, K. J. Bai, R. C. Ertekin, and W. C. Webster. A derivation of the green-naghdi equations for irrotational flows. *J. Eng. Math.*, 40:17–42, 2001.

[150] I. Kimura and Y. Shimizu. Refinement of MPS method for simulating avalanches. *Journal of Japan Society of Civil Engineers Ser B1 (Hydraulic Engineering)*, 70(4):I_433–I_438, 2014.

[151] S. Kolzenburg, D. Giordano, A. Di Muro, and D. B. Dingwell. Equilibrium viscosity and disequilibrium rheology of a high magnesium basalt from piton de la fournaise volcano. *Ann. Geophys.*, 625, 2018. doi: 10.1029/2011GC003606.

[152] A. D. Korawan, S. Soeparman, W. Wijayanti, and D. Widhiyanuriyawan. 3d numerical and experimental study on paraffin wax melting in thermal storage for the nozzle-and-shell, tube-and-shell, and reducer-and-shell models. *Modelling and Simulation in Engineering*, 2017, 2017. doi: 10.1155/2017/9590214.

[153] P. Kosky, R. Balmer, W. Keat, and G. Wise. *Exploring Engineering*. Academic Press, 3 edition, 2013. ISBN 9780124158917.

[154] I. M. Krieger and T. J. Dougherty. A mechanism for non-newtonian flow in suspensions of rigid spheres. *Transactions of the Society of Rheology*, 3(1):137–152, 1959.

[155] J. Kubanek, J. A. Richardson, S. J. Charbonnier, and L. J. Connor. Lava flow mapping and volume calculations for the 2012–2013 Tolbachik, Kamchatka, fissure eruption using bistatic TanDEM-X InSAR. *Bulletin of Volcanology*, 77(12):106, 2015. doi: 10.1007/s00445-015-0989-9.

[156] A. Kurganov and D. Levy. A third-order semidiscrete central scheme for conservation laws and convection-diffusion equations. *SIAM J. Sci. Comput.*, 22:1461–1488, 200.

[157] A. Kurganov and D. Levy. Central-upwind schemes for the Saint-Venant system. *ESAIM: Mathematical Modelling and Numerical Analysis*, 36(3):397–425, 2002.

[158] A. Kurganov and G. Petrova. A second-order well-balanced positivity preserving central-upwind scheme for the Saint-Venant system. *Commun. Math. Sci.*, 5(1):133–160, 2007.

[159] A. Kurganov and E. Tadmor. New high-resolution central schemes for nonlinear conservation laws and convection–diffusion equations. *Journal of Computational Physics*, 160: 241–282, 2000.

[160] A. Kurganov, S. Noelle, and G. Petrova. Semidiscrete central-upwind schemes for hyperbolic conservation laws and Hamilton-Jacobi equations. *SIAM*, 23(3):707–740, 2001.

[161] G. La Spina and M. d. M. Vitturi. High-resolution finite volume central schemes for the sint-venant system. *SIAM Journal on Scientific Computing*, 34:861–880, 2012.

[162] P. D. Lax and B. Wendroff. Systems of conservation laws. *Comm. Pure Appl. Math.*, 13: 217–237, 1960.

[163] R. Lecuna, F. Delgado, A. Ortiz, P. B. Castro, I. Fernandez, and C. J. Renedo. Thermal-fluid characterization of alternative liquids of power transformers: a numerical approach. *IEEE Transactions on Dielectrics and Electrical Insulation*, 22(5):2522–2529, 2015.

[164] E. Lev and M. R. James. The influence of cross-sectional channel geometry on rheology and flux estimates for active lava flows. *Bulletin of Volcanology*, 76(829), 2014. doi: 10.1007/s00445-014-0829-3.

[165] E. Lev, M. Spiegelman, R. J. Wysocki, and J. Karson. Investigating lava flow rheology using video analysis and numerical flow models. *Journal of Volcanology and Geothermal Research*, 247–248:62–73, 2012.

[166] E. Lev, M. Spiegelman, R. J. Wysocki, and J. A. Karson. Investigating lava flow rheology using video analysis and numerical flow models. *Journal of Volcanology and Geothermal Research*, 2012.

[167] R. J. LeVeque. Balancing source terms and flux gradients in high-resolution Godunov methods: the quasi-steady wave-propagation algorithm. *J. Comput. Phys.*, 146:346–365, 1998.

[168] R. J. LeVeque. *Finite volume methods for hyperbolic problems*. Cambridge Texts in Applied Mathematics, 2004.

[169] S. Li and C. J. Duffy. Fully-coupled modeling of shallow water flow and pollutant transport on unstructured grids. *Procedia Environmental Sciences*, 13(3):2098–2121, 2012.

[170] Q. Liang and A. G. L. Borthwick. Adaptive quadtree simulation of shallow flows with wet–dry fronts over complex topography. *Computers & Fluids*, 38(2):221–234, 2009.

[171] Z. Lilek. Ein Finite-Volumen Verfahren zur Berechnung von inkompressiblen und kompressiblen Stromungen in komplexen Geometrien mit beweglichen Randern und freien OberflEhen. In *Dissertation*. University of Hamburg, Germany, 1995.

[172] J. R. Lister. Viscous flows down an inclined plane from point and line sources. *Journal of Fluid Mechanics*, 242:631–653, 1992.

[173] X. Liu. A robust numerical model for shallow water governing solute transport with wet/dry interfaces. *Comput. Methods Appl. Mech. Engrg.*, 351:85–108, 2019.

[174] X. Liu, J. Albright, Y. Epshteynb, and A. Kurganov. Well-balanced positivity preserving central-upwind scheme with a novel wet/dry reconstruction on triangular grids for the Saint-Venant system. *Journal of Computational Physics*, 374:213–236, 2018.

[175] R. D. Lonsdale. An algebraic multigrid scheme for solving the Navier–Stokes equations on unstructured meshes. *International Journal of Numerical Methods for Heat & Fluid Flow*, 3(1):3–14, 1993. doi: 10.1108/eb017512.

[176] S. C. Loughlin. Dynamic Risk at Fuego Volcano: Communities living in a post-eruption but still persistently active context. 2018. URL http://119.78.100.173/C666/handle/2XK7JSWQ/87263.

[177] J. C. Luke. A variational principle for a fluid with a free surface. *Journal of Fluid Mechanics*, 27:395–397, 1967.

[178] J. Y. Luo, R. I. Issa, and A. D. Gosman. Prediction of impeller induced flows in mixing vessels using multiple frames of reference. *IChemE Symposium Series*, 136:549–556, 1994.

[179] F. Lyard, F. Lefevre, T. Letellier, and O. Francis. Modelling the global ocean tides: modern insights from FES2004. *Ocean dynamics*, 56(5-6):394–415, 2006.

[180] J. N. Lyness. Numerical algorithms based on the theory of complex variable. In *Proceedings of the 1967 22nd national conference*, pages 125–133. ACM, 1967.

[181] J. N. Lyness and C. B. Moler. Numerical differentiation of analytic functions. *SIAM Journal on Numerical Analysis*, 4(2):202–210, 1967.

[182] C. L. Mader. *Numerical modeling of water waves*. CRC press, 2004.

[183] R. Manning, J. P. Griffith, T. F. Pigot, and L. F. Vernon-Harcourt. *On the flow of water in open channels and pipes*. 1890.

[184] D. L. Marchisio and R. O. Fox. *Computational models for polydisperse particulate and multiphase systems*. 2013. ISBN 9780521858489.

[185] J. R. R. A. Martins, I. M. Kroo, and J. J. Alonso. An automated method for sensitivity analysis using complex variables. *AIAA paper 2000-0689, 38th Aerospace Sciences Meeting*, 2000.

[186] J. R. R. A. Martins, P. Sturdza, and J. J. Alonso. The complex-step derivative approximation. *ACM Transactions on Mathematical Software*, 29(3):245–262, 2003.

[187] J. Martí Molist, L. Becerril, and S. Bartolini. Using VOLCANBOX to conduct hazard assessment in monogenetic volcanic fields. In *Abstracts volume 7th International Maar Conference*, pages 182–183, 2018.

[188] MDPI, editor. *Two-dimensional shallow-water model with porosity for urban flood modeling*, 2018.

[189] E. T. W. Mei, F. Lavigne, A. Picquout, E. de Bélizal, D. Brunstein, D. Grancher, J. Sartohadi, N. Cholikc, and C. Vidal. Lessons learned from the 2010 evacuations at Merapi volcano. *Journal of Volcanology and Geothermal Research*, 261(1):348–365, 2013. doi: 10.1016/j.jvolgeores.2013.03.010.

[190] M. Meier, G. Yadigaroglu, and B. L. Smith. A novel technique for including surface tension in plic-vof methods. *European J. Mech. B-fluids*, 21(1):61–73, 2002.

[191] T. Menard, S. Tanguy, and A. Berlemont. Coupling level set/vof/ghost fluid methods: Validation and application to 3d simulation of the primary breakup of a liquid jet. *Int. J. Multiphase Flow*, 33(5):510–524, 2007.

[192] B. D. Michel, B. Piar, J. C. Babik, F. Latche, G. Guillard, and C. D. Pascale. Synthesis of the validation of croco v1 spreading code. *Proceedings of the OECD Workshop on Ex-Vessel Debris Coolability*, 6475:235–245, 2000.

[193] C. Min. On reinitializing level set functions. *Journal of Computational Physics*, pages 2764–2772, 2010.

[194] M. F. Modest. *Radiative heat transfer.* McGraw-Hill, New York, 1993.

[195] S. Muzaferija. *Adaptive Finite Volume method for flow prediction using unstructured meshes and multigrid approach.* PhD thesis, Imperial College, University of London, 1994.

[196] S. Márquez Damián. *An extended mixture model for the simultaneous treatment of short and long scale interfaces.* PhD thesis, Universidad Nacional del Litoral, 2013.

[197] Y. Nakamura and K. Aoki. The 1977 eruption of Nyiragongo volcano, eastern Africa, and chemical composition of the ejecta. *Bulletin of the Volcanological Society of Japan*, 25(1): 17–32, 1980. doi: 10.18940/kazanc.25.1_17.

[198] S. Noelle, N. Pankratz, G. Puppo, and J. Natvig. Well-balanced finite volume schemes of arbitrary order of accuracy for shallow water flows. *J. Comput. Phys.*, 213:474–499, 2006.

[199] P. of the OECD Workshop on Ex-Vessel Debris Coolability, editor. *Simulations of core melt spreading with lava: theoretical background and status of validation*, 2000.

[200] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables.* Cambridge university, 1987.

[201] S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *J . Comput. Phys.*, 79:12–49, 1988.

[202] M. Ostrogradskij. Première note sur la théorie de la chaleur. *Mémoires de l'Académie impériale des sciences de St. Pétersbourg*, 6:129–133, 1831.

[203] P. Papale. *Volcanic hazards, risks and disasters.* Amsterdam ; Boston : Elsevier, 2015.

[204] L. Pareschi and G. Russo. Implicit-Explicit Runge-Kutta schemes for stiff system of differential equations. In *Recent trends in numerical analysis*, pages 269–288. Nova Science, 2000.

[205] A. Parmigiani, C. Huber, O. Bachmann, and B. Chopard. Pore-scale mass and reactant transport in multiphase porous media flows. *Journal of Fluid Mechanics*, 686:40–76, 2011. doi: 10.1017/jfm.2011.268.

[206] A. Parmigiani, J. Latt, M. B. Begacem, and B. Chopard. A lattice Boltzmann simulation of the Rhone river. *International Journal of Modern Physics C*, 24, 2013. doi: 10.1142/S0129183113400081.

[207] S. V. Patankar. *Numerical heat transfer and fluid flow.* McGraw-Hill, NY, 1980.

[208] S. V. Patankar and D. B. Spadling. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *International Journal of Heat and Mass Transfer*, 15:1787–1806, 1972.

[209] M. R. Patrick, J. Dehn, and K. Dean. Numerical modeling of lava flow cooling applied to the 1997 Okmok eruption: Approach and analysis. *J. Geophys. Res.*, 109, 2004. doi: 10.1029/2003JB002537.

[210] M. R. Patrick, H. R. Dietterich, J. J. Lyons, A. K. Diefenbach, C. Parcheta, K. R. Anderson, A. Namiki, I. Sumita, B. Shiro, and J. P. Kauahikaua. Cyclic lava effusion during the 2018 eruption of Kīlauea Volcano. *Science*, 366, 2019. doi: 10.1126/science.aay9070.

[211] D. H. Peregrine. Equations for water waves and the approximation behind them. 1972.

[212] M. Peric. *A Finite Volume method for the prediction of three-dimensional fluid flow in complex ducts.* PhD thesis, Imperial College, University of London, 1985.

[213] B. Perthame and C. Simeoni. A kinetic scheme for the Saint-Venant system with a source term. *Calcolo*, 38:201–231, 2001.

[214] A. A. Petrov. Variational statement of the problem of liquid motion in a container of finite dimensions. *Prikl. Math. Mekh.*, 28:917–922, 1964.

[215] D. Pieri and S. Baloga. Eruption rate, area, and length relationships for some Hawaiian lava flows. *J. Volcanol. Geotherm. Res.*, 30:29–45, 1986.

[216] H. Pinkerton and R. S. J. Sparks. Field measurements of the rheology of lava. *Nature*, 276:383–385, 1978.

[217] D. Pokrajac, S. Venuleo, and M. J. Franca. Depth-averaged momentum equation for gravity currents with varying density: coefficient in pressure term. *Journal of Hydraulic Research*, 56:3:424–430, 2018. doi: 10.1080/00221686.2017.1335245.

[218] S. Popinet and S. Zaleski. A front-tracking algorithm for accurate representation of surface tension. *Int. J. Numer. Meth. in Fluids*, 30(6):775–793, 1999.

[219] L. Prandtl. über Flüssigkeitsbewegung bei sehr kleiner Reibung. *Verhandl III, Intern. Math. Kongr. Heidelberg*, 2:484–491, 1904.

[220] I. Proceedings World Geothermal Congress 2010, Bali, editor. *Determination of Thermal Conductivity of Coarse and Fine Sand Soils*, 2010.

[221] J. N. Procter, S. J. Cronin, I. C. Fuller, M. Sheridan, V. E. Neall, and H. Keys. Lahar hazard assessment using Titan2D for an alluvial fan with rapidly changing geomorphology: Whangaehu River, Mt. Ruapehu. *Geomorphology*, 116(1-2):162–174, 2010.

[222] E. Rader, L. Vanderkluysen, and A. Clarke. The role of unsteady effusion rates on inflation in long-lived lava flow fields. *Earth and Planetary Science Letters*, 477:73–83, 2017.

[223] G. D. Raithby, W. X. Xu, and G. D. Stubley. Prediction of incompressible free surface flows with an element-based finite volume method. *Comput. Fluid Dynamics J.*, 4:353–371, 1995.

[224] O. Reynolds. An experimental investigation of the circumstances which determine whether the motion of water shall be direct or sinuous, and of the law of resistance in parallel channels. *Philosophical Transactions of the Royal Society*, (174):935–982, 1883. doi: 10.1098/rstl.1883.0029.

[225] O. Reynolds. *Papers on Mechanical and Physical Subjects*, volume 3. 1903.

[226] L. Rezzolla. Numerical methods for the solution of partial differential equations. 2011.

[227] C. M. Rhie and W. L. Chow. Numerical study of the turbulent flow past an airfoil with trailing edge separation. *AIAA Journal*, 21:1525–1532, 1983. doi: 10.2514/3.8284.

[228] M. Rhéty, A. J. L. Harris, N. Villeneuve, L. Gurioli, E. Médard, M. O. Chevrel, and P. Bachèlery. A quantitative approach for evaluating lava flow simulation reliability: LavaSIM code applied to the 2001 Etna eruption. *Geochemistry, Geophys. Geosystems*, 10 (9), 2009. doi: 10.1029/2009GC002426.

[229] M. Rhéty, A. J. L. Harris, N. Villeneuve, L. Gurioli, E. Médard, M. O. Chevrel, and P. Bachèlery. A comparison of cooling-limited and volume-limited flow systems: Examples from channels in the Piton de la Fournaise April 2007 lava-flow field. *Geochemistry, Geophys. Geosystems*, 18(9):3270–3291, 2017. doi: 10.1002/2017GC006839.

[230] P. Richardson and L. Karlstrom. The multi-scale influence of topography on lava flow morphology. *Bulletin of Volcanology*, 81(21), 2019. doi: 10.1007/s00445-019-1278-9.

[231] N. Richter, M. Favalli, E. de Zeeuw-van Dalfsen, A. Fornaciai, R. M. da Silva Fernandes, N. M. Pérez, J. Levy, S. V. Silva, and T. R. Walter. Lava flow hazard at Fogo Volcano, Cabo Verde, before and after the 2014–2015 eruption. *Nat. Hazards Earth Syst. Sci.*, 16: 1925–1951, 2016.

[232] P. Rizzoli, M. Martone, C. Gonzalez, C. Wecklich, D. Borla Tridon, B. Bräutigam, M. Bachmann, D. Schulze, T. Fritz, M. Huber, B. Wessel, G. Krieger, M. Zink, and A. Moreira. Generation and performance assessment of the global TanDEM-X digital elevation model. *ISPRS Journal of Photogrammetry and Remote Sensing*, 132:119–139, 2017. doi: 10.1016/j.isprsjprs.2017.08.008.

[233] J. Roenby, H. Bredmose, and H. Jasak. A Computational Method for Sharp Interface Advection. *Royal Society Open Science*, 3(11), 2016. doi: 10.1098/rsos.160131.

[234] J. Roenby, B. Eltard Larsen, H. Bredmose, and H. Jasak. A New Volume-of-Fluid Method in OpenFOAM. In *MARINE 2017 Computational Methods in Marine Engineering VII*, pages 266–278. International Center for Numerical Methods in Engineering, 2017.

[235] H. Rouse. Nomogram for the settling velocity of spheres. *Division of Geology and Geography, Exhibit D of the Report of the Commission on Sedimentation*, pages 57–64, 1937.

[236] M. Rudman. Volume-tracking methods for interfacial flow calculations. *Int.l J. Numer. Meth. Fluids*, 24(7):671–691, 1997.

[237] G. Russo. Central schemes for balance laws. In Freistühler H., Warnecke G., editor, *Hyperbolic Problems: Theory, Numerics, Applications*, ISNM International Series of Numerical Mathematics, 2001. doi: 10.1007/978-3-0348-8372-6_35.

[238] G. Russo. Central schemes for conservation laws with application to shallow water equations. In R. G. e. Rionero S., editor, *Trends and Applications of Mathematics to Mechanics*, pages 225–246. The organization, Springer, 2005.

[239] D. Rust and M. Neri. The boundaries of large-scale collapse on the flanks of Mount Etna, Sicily, in *Volcano Instability on the Earth and Other Planets,* edited by W. M. McGuire, A. P. Jones, and J. Neuberg. *Geol. Soc. London, Spec. Publ.*, 110:193–208, 1996.

[240] P. Saramito, C. Smutek, and B. Cordonnier. Numerical modeling of shallow non-Newtonian flows: part I. the 1D horizontal dam break problem revisited. *International journal of numerical analysis & modeling, Series B*, 4:283–298, 2013.

[241] H. Schlichting. *Boundary-layer theory.* McGraw Hill, 1979.

[242] G. E. Schneider and M. J. Raw. Control volume finite-element method for heat transfer and fluid flow using colocated variables - 1. computational procedure. *Numerical Heat Transfer*, 11:363–390, 1987. doi: 10.1080/10407788708913560.

[243] Y. Shah and J. Pearson. Stability of non-isothermal flow in channels: III. Temperature-dependent pawer-law fluids with heat generation. *Chem. Eng. Sci.*, 29:1485–1493, 1974.

[244] M. F. Sheridan, A. J. Stinton, A. Patra, E. B. Pitman, A. Bauer, and C. C. Nichita. Evaluating Titan2D mass-flow model using the 1963 Little Tahoma peak avalanches, Mount Rainier, Washington. *Journal of volcanology and geothermal research*, 139(1-2):89–102, 2005.

[245] E. Shirani, N. Ashgriz, and J. Mostaghimi. Interface pressure calculation based on conservation of momentum for front capturing methods. *J. Comp. Phys.*, 203(1):154–175, 2005.

[246] K. Sieron, D. Ferrés, C. Siebe, R. Constantinescu, L. Capra, C. Connor, L. Connor, G. Groppelli, and K. G. Zuccolotto. Ceboruco hazard map: part II - modeling volcanic phenomena and construction of the general hazard map. *Nat. Hazards*, 96:893–933, 2019. doi: 10.1007/s11069-019-03577-5.

[247] B. Spindler and J. M. Veteau. The simulation of melt spreading with THEMA code: part 1: model, assessment strategy and assessment against analytical and numerical solutions. *Nuclear Engineering and Design*, 236:415–424, 2006. doi: 10.1016/j.nucengdes.2005.09.009.

[248] W. Squire and G. Trapp. Using complex variables to estimate derivatives of real functions. *Siam Review*, 40(1):110–112, 1998.

[249] R. B. Stothers. The great Tambora eruption of 1815 and is aftermath. *Science*, 224: 1191–1198, 1984. doi: 10.1126/science.224.4654.1191.

[250] D. K. Syahbana, K. Kasbani, G. Suantika, O. Prambada, A. S. Andreas, U. B. Saing, S. L. Kunrat, S. Andreastuti, M. Martanto, E. Kriswati, Y. Suparman, H. Humaida, S. Ogburn, P. J. Kelly, J. Wellik, H. M. N. Wright, J. D. Pesicek, R. Wessels, C. Kern,

M. Lisowski, A. Diefenbach, M. Poland, F. Beauducel, J. Pallister, R. G. Vaughan, and J. B. Lowenstern. The 2017–19 activity at Mount Agung in Bali (Indonesia): Intense unrest, monitoring, crisis response, evacuation, and eruption. *Scientific Reports*, 9(8848), 2019. doi: 10.1038/s41598-019-45295-9.

[251] H. S. Tang and D. M. Kalyon. Estimation of the parameters of Herschel-Bulkley fluid under wall slip using a combination of capillary and squeeze flow viscometers. *Rheol Acta*, (43):80–88, 2004. doi: 10.1007/s00397-003-0322-y.

[252] S. Tarquini and M. Favalli. Mapping and downflow simulation of recent lava flow fields at mount etna. *Journal of Volcanology and Geothermal Research*, 204:27–39, 2011.

[253] S. Tarquini and M. Favalli. Uncertainties in lava flow hazard maps derived from numerical simulations: the case study of Mount Etna. *Journal of Volcanology and Geothermal Research*, 260:90–102, 2013.

[254] S. Tarquini, M. Favalli, M. Pfeffer, M. de' Michieli Vitturi, S. Barsotti, G. Pedersen, B. Óladóttir, and E. H. Jensen. Assessing the impact of lava flows during the 2020 unrest of the Svartsengi volcanic system on the Reykjanes peninsula, Iceland. 2020. doi: 10.30437/GEOMORPHOMETRY2020.

[255] H. Tennekes and J. L. Lumley. *A First Course in Turbulence*. MIT Press, 1972.

[256] T. Thordarson and S. Self. Atmospheric and environmental effects of the 1783–1784 Laki eruption: a review and reassessment. *Journal of Geophysical Research*, 108, 2003. doi: 10.1029/2001JD002042.

[257] J. L. Thé, G. D. Raithby, and G. D. Stubley. Surface-adaptive finite-volume method for solving free-surface flows. *Numer. Heat Transfer*, 26:367–380, 1994.

[258] E. F. Toro. The dry bed problem in shallow water flows. *College of aeronautics, Report No. 9007*, 1990.

[259] S. W. R. Tsang and J. M. Lindsay. Lava flow crises in inhabited areas part I: lessons learned and research gaps related to effusive, basaltic eruptions. *Journal of Applied Volcanology*, 9, 2020. doi: 10.1186/s13617-020-00096-.

[260] J. P. Van Doormal and G. D. Raithby. Enhancements of the SIMPLE method for predicting incompressible fluid flows. *Numer. Heat Transfer*, 7:147–163, 1984.

[261] S. P. Vanka. Block-implicit multigrid solution of Navier–Stokes equations in primitive variables. *Journal of Computational Physics*, 65:138–158, 1986.

[262] V. A. Vanoni. Nomogram for the settling velocity of spheres. *Transportation of suspended sediment by water. Transactions of ASCE*, 111:67–133, 1946.

[263] A. Vicari, H. Alexis, C. Del Negro, M. Coltelli, M. Marsella, and C. Proietti. Modeling of the 2001 lava flow at Etna volcano by a cellular automata approach. *Environmental Modelling & Software*, 22(10):1465–1471, oct 2007. doi: 10.1016/j.envsoft.2006.10.005.

[264] V. Vukcevic. *Numerical modelling of coupled potential and viscous flow for marine applications*. PhD thesis, University of Zagreb, Zagreb, Croatia, 2016.

[265] V. Vukcevic, H. Jasak, and S. Malenica. Decomposition model for naval hydrodynamic applications, part i: Computational method. *Ocean Eng.*, 121:37–46, 2016.

[266] G. Wadge, P. A. V. Young, and I. J. McKendrick. Mapping lava flow hazards using computer simulation. *Journal of Geophysical Research: Solid Earth*, 99:489–504, 1994.

[267] G. Walker. Thickness and viscosity of Etnean lavas. *Nature*, 213(5075):484–485, feb 1967. doi: 10.1038/213484a0.

[268] G. Walker. Lengths of lava flows. *Philosophical Transactions for the Royal Society of London. Series A, Mathematical and Physical Sciences*, 274:107–116, 1973.

[269] Y. Wang, K. Hutter, and S. P. Pudasaini. The Savage-Hutter theory: a system of partial differential equations for avalanche flows of snow, debris, and mud. *ZAMM*, 84(8):507–527, 2004.

[270] M. Wantim, M. Kervyn, G. Ernst, M. del Marmol, C. Suh, and P. Jacobs. Numerical experiments on the dynamic of channelised lava flows at Mount Cameroon volcano with the FLOWGO thermo-rheological model. *Journal of Volcanology and Geothermal Research*, 253:35–53, 2013.

[271] R. Wemmenhove, R. Luppes, A. E. P. Veldman, and T. Bunnik. Numerical simulation of hydrodynamic wave loading by a compressible two-phase flow method. *Computers and Fluids*, 114:218–231, 2015.

[272] G. B. Whitham. *Linear and nonlinear waves*. John Wiley, New York, 1974.

[273] D. L. Williamson, J. B. Drake, J. J. Hack, R. Jakob, and P. N. Swarztrauber. A standard test set for numerical approximations to the shallow water equations in spherical geometry. *Journal of Computational Physics*, 102(1):211–224, 1992.

[274] A. Wirth. A guided tour through physical oceanography. *Physical Oceanography*, 2015.

[275] R. Wittmaak. Corflow: a code for the numerical simulation of free-surface flow. *Nuclear Technology*, 119:158–180, 1997.

[276] X. Xia and Q. Liang. A new depth-averaged model for flow-like landslides over complex terrains with curvatures and steep slopes. *Engineering Geology*, 234:174–191, 2018. doi: 10.1016/j.enggeo.2018.01.011.

[277] Y. Xing and C. W. Shu. High order finite difference WENO schemes with the exact conservation property for the shallow water equations. *J. Comput. Phys.*, 208:206–227, 2005.

[278] Y. Xing and C. W. Shu. A new approach of high order well-balanced finite volume WENO schemes and discontinuous Galerkin methods for a class of hyperbolic systems with source terms. *Commun. Comput. Phys*, 1:100–134, 2006.

[279] A. Yahalom and D. Lynden-Bell. Simplified variational principles for barotropic magneto-hydrodynamics. *J. Fluid. Mech.*, 607:235–265, 2008.

[280] P. Young and G. Wadge. FLOWFRONT: Simulation of a lava flow. *Computers & Geosciences*, 16(8):1171–1191, jan 1990. doi: 10.1016/0098-3004(90)90055-X.

[281] V. Zago, G. Bilotta, A. Cappello, R. A. Dalrymple, L. Fortuna, G. Ganci, A. Herault, and C. D. Negro. SPH model for the simulation of lava-buildings interactions. *Journal of Volcanology and Geothermal Research*, 349:323–334, 2018. doi: 10.1016/j.jvolgeores.2017.11.016.

[282] V. Zago, G. Bilotta, A. Cappello, R. A. Dalrymple, L. Fortuna, G. Ganci, A. Herault, and C. D. Negro. Preliminary validation of lava benchmark tests on the GPUSPH particle engine. *Annals of Geophysics*, 62, 2019.

[283] V. E. Zakharov. Stability of periodic waves of finite amplitude on the surface of a deep fluid. *J.Appl.Mech.Tech.Phys.*, 9:1990–1994, 1968.

[284] A. Ziv. Relative distance-an error measure in round-off error analysis. *Mathematics of Computation*, 39(160):563–569, 1982.